Universidad Autónoma
de Madrid

Escuela Politécnica Superior

Dpto. de Ingeniería Informática

Doctorado en Ingeniería Informática y Telecomunicación

Tesis doctoral

Algoritmos bio-inspirados para la detección de comunidades dinámicas en redes complejas

Autor

ÁNGEL PANIZO LLEDOT

Tutores

Dr. D. DAVID CAMACHO FERNÁNDEZ
Dra. Dña. GEMA BELLO ORGAZ

Enero 2022

Departamento: Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid (UAM)
España

Tesis doctoral: "Algoritmos bio-inspirados para la detección de comunidades dinámicas en redes complejas"

Autor: **Ángel Panizo LLedot**
Ingeniero en Informática
Universidad Complutense de Madrid, Spain

Tutores: **David Camacho Fernández**
Doctor Ingeniero en Informática
Universidad Politécnica de Madrid, Spain
Gema Bello Orgaz
Doctora Ingeniera en Informática
Universidad Politécnica de Madrid, Spain

Año: 2022

Comité: Presidente: Sancho Salcedo Sanz
Secretario: Álvaro Manuel Ortigosa Juarez
Vocal 1: Ricardo Aler Mur
Vocal 2: Alejandro Martín García
Vocal 3: Cristian Ramírez Atencia

Resumen

Esta tesis se centra en el **desarrollo de algoritmos meta-heurísticos poblacionales eficientes aplicables a dominios de redes dinámicas, es decir, que evolucionan en el tiempo**. Específicamente, en el desarrollo de Genetic Algorithm (Algoritmos genéticos) (GA) eficientes para la detección de comunidades dinámicas. En pocas palabras, podemos considerar al Community Detection Problem (Problema de detección de comunidades) (CDP) como el proceso de hacer *clustering* sobre redes, es decir, dada una red compuesta por nodos o entidades y aristas o conexiones entre estas entidades, el CDP consiste en hacer grupos de nodos de tal manera que los miembros de un mismo grupo contenga más conexiones entre ellos que entre el resto de nodos de la red.

Las meta-heurísticas bio-inspiradas ofrecen un compromiso excelente entre potencia, flexibilidad y escalabilidad. Su mejor ventaja es que son capaces de generar soluciones con una calidad aceptable en un tiempo de ejecución acotado, cosa que los métodos clásicos no pueden ofrecer. Cuando un problema crece hasta tamaños dónde los métodos clásicos dejan de ser aplicables, ya sea por tiempo o memoria, un GA (o en su defecto cualquier meta-heurística bio-inspirada) son candidatos excelentes para obtener soluciones con una calidad aceptable, consumiendo una cantidad de recursos asumible. Los GAs son una de las familias más populares que forman parte del conjunto de algoritmos meta-heurísticos bio-inspirados. Dichos algoritmos han sido aplicados en centenares de dominios de forma satisfactoria, incluido el CDP.

Aplicar un GA a un entorno dinámico de manera satisfactoria no es trivial y es necesario definir mecanismos extra. Los GAs son heurísticos poblacionales, dado que durante el proceso de ejecución no existe una única solución que se va mejorando, sino que existe un conjunto de ellas. Cuando trabajamos en un entorno dinámico y aparecen cambios, es posible que la población deje de ser capaz de encontrar soluciones satisfactorias al problema. Una solución trivial a este problema es descartar la población actual cuando aparezca un cambio y crear una nueva completamente de cero. Sin embargo, dicha estrategia descarta todo trabajo pasado, lo que la convierte en ineficiente y no aplicable al mundo real.

Esta tesis se ha centrado, por lo tanto, en la aplicación de GAs para la detección de comunidades dinámicas en redes, en concreto, a estudiar aquellas técnicas que permiten reducir el tiempo de ejecución, en aras de hacer estos algoritmos aplicables al mundo real. Se identificaron principalmente dos escollos para aplicar un GA al CDP dinámico: (I) la necesidad de adaptar los individuos a los cambios en la topología de la red para evitar la aparición de soluciones inválidas; (II) la necesidad de adaptar las poblaciones a cambios en la red para evitar que pierdan capacidades.

Las principales contribuciones de la tesis doctoral pueden resumirse en:

- Respecto a la aparición de soluciones inválidas, se han propuesto tres operadores capaces de reparar un individuo, es decir, transformar una solución inválida en una solución válida de nuevo. A la hora de realizar el proceso de reparación, los mecanismos propuestos son capaces de aprovechar el conocimiento pasado, de tal manera que apenas se pierda información durante el proceso de reparación.

Para probar la validez de dichos mecanismos se ha llevado un caso de uso en el que se integraron en un GA concreto para la detección de comunidades dinámicas y se probó el rendimiento de dicho GA usando una batería de pruebas compuesta por 66 redes sintéticas y dos reales.

- En cuanto al segundo problema abordado, en el estado del arte se pueden encontrar varias técnicas que permiten adaptar una población a un cambio del entorno de tal manera que no pierda capacidades y, entre ellas, las técnicas de inmigrantes son de las más populares. Aunque implementar dichas técnicas en un algoritmo genético es un proceso sencillo, su posterior ajuste no lo es tanto, ya que se tienen que configurar una serie de parámetros que pueden afectar de manera crítica al rendimiento del algoritmo. En aras de hacer este proceso más sencillo, esta tesis propone una nueva metodología que permite evaluar estrategias de inmigrantes en GAs en base a tres dimensiones diferentes: Calidad, Estabilidad y Velocidad. La metodología se diseñó con un propósito doble: por un lado, evaluar si una determinada estrategia de inmigrantes está mejorando o perjudicando alguna de las dimensiones mencionadas y, por otro, dar información sobre lo que podría estar ocurriendo en el caso de que alguna de las dimensiones se vea afectada negativamente. Finalmente, y para probar la validez de esta metodología se llevó a cabo un caso de uso en el que se evaluaron tres de las estrategias de inmigrantes más populares usando 18 redes dinámicas sintéticas.
- Se han realizado un conjunto de validaciones y análisis experimentales respecto a la utilización de GAs en el dominio de la deteccción de comunidades dinámicas que ha permitido establecer:
  1. Qué método, o métodos, son los más eficientes para adaptar los individuos de un GA diseñado para la detección de comunidades a los cambios del entorno y así evitar la aparición de soluciones inválidas.
  2. Cuál de las estrategias presentes en el estado del arte para adaptar la población de un GA a cambios en el entorno es la que mejor funciona para la versión dinámica del CDP y qué ventajas e inconvenientes presenta.
  3. Del conocimiento obtenido durante la realización de esta tesis, se ha tratado de establecer cómo podrían generalizarse los resultados obtenidos a otros dominios, es decir, cómo puede elegirse la mejor estrategia para adaptar las poblaciones de un GA en otros problemas dinámicos ajenos al del CDP.

Abstract

This thesis revolves around the **development of efficient population meta-heuristic algorithms applicable to dynamic, i.e. time-evolving, networks domain**. Specifically, this work is focused in the design and development of new efficient Genetic Algorithms (GAs) for the detection of dynamic communities in networks. In a nutshell, we can consider the Community Detection Problem (CDP) as the process of doing *clustering* on networks. In other words, given a network composed of nodes or entities and edges or links between these entities, the CDP consists of grouping nodes in such a way that the members of a group contain more links between them than between the rest of the nodes of the network.

Bio-inspired meta-heuristics offer an excellent trade-off between power, flexibility and scalability. Their best advantage is that they are able to generate solutions with acceptable quality in a limited execution time, something that classical methods cannot deliver. When a problem grows to sizes where classical methods are no longer applicable, either by time or memory, a GA (or any other bio-inspired meta-heuristic method) is an excellent approach to obtain solutions with an acceptable quality, consuming a manageable amount of resources. The GAs are one of the most popular families of bio-inspired meta-heuristic algorithms and they have been successfully applied in hundreds of domains, including the CDP [81].

Applying a GA to a dynamic environment in a successful way is not trivial and extra mechanisms are required. The GAs are population heuristics, that is, they improve a set of solutions instead of improving a single one. When working in a dynamic environment, changes can disrupt a population in such a way that it is no longer able to find satisfactory solutions. A trivial solution to this problem is to discard the current population when a change occurs and creates a new one completely from scratch. However, such a strategy discards all previous work and computational efforts, making it inefficient and making it more difficult for these techniques to be applied to real problems.

This thesis has therefore focused on the application of GAs for the detection of dynamic communities in networks, in particular, to study those techniques that allow to reduce the execution time in order to make these algorithms applicable to the real world. Two main challenges were identified for the application of a GA to dynamic CDP: (I) The need to adapt individuals to changes in the network topology in order to avoid the occurrence of invalid solutions; (II) The need to adapt populations to changes in the network to prevent them from losing capabilities.

Particularly, the main contributions of the doctoral thesis can be summarised as follows:

- Regarding the occurrence of invalid solutions, three operators have been proposed that are capable of repairing an individual, i.e. transforming an invalid

solution into a valid solution again. When carrying out the repairing process, the proposed operators are able to take advantage of past knowledge, so that hardly any information is lost during this process. In order to test the validity of these operators, a use case has been conducted in which they were integrated into a particular GA for detecting dynamic communities and the performance of this GA was evaluated using a benchmark consisting of 66 synthetic networks and two real ones.

- Regarding the second problem addressed, the state of the art includes several techniques that allow a population to adapt to changes in the environment in such a way that it does not lose capabilities. Among those techniques, immigrant schemes are among the most popular. Although implementing these techniques in a GA is a simple process, their subsequent adjustment is not so simple, since a series of parameters have to be configured that can critically affect the performance of the algorithm. In order to make this process simpler, this thesis proposes a new methodology that allows the evaluation of immigrant schemes in GAs based on three different dimensions: Quality, Stability and Speed. The methodology was designed with a twofold purpose in mind. On the one hand, to assess whether a given immigrant strategy is improving or hindering any of the mentioned dimensions. On the other hand, to provide information on what might be happening in the case that any of the dimensions is negatively affected. Finally, to test the adequacy of this methodology, a use case was carried out in which three of the most popular immigrant strategies were evaluated using 18 synthetic dynamic networks.
- A set of experimental analyses and empirical validations have been carried out on the use of GAs in the domain of dynamic community detection, which has allowed us to establish:
  1. Which method, or methods, are the most efficient to adapt the individuals of a GA designed for community detection to changes in the environment and thus avoid the occurrence of invalid solutions.
  2. Which of the strategies available in the state of the art for adapting the population of a GA to changes in the environment performs best for the dynamic version of the CDP and what are the advantages and disadvantages of this strategy.
  3. From the knowledge obtained during the development of this thesis, we have tried to establish how the results obtained could be generalised to other domains, i.e. we have tried to establish how the best strategy can be chosen to adapt the populations of an GA in other dynamic problems outside the CDP.

Agradecimientos

A Gema y David por guiarme durante todos los años que ha llevado realizar esta tesis.

A mis padres, Ángel y Laura, por haberme apoyado siempre en las decisiones que he tomado.

A mi familia y amigos por estar siempre a mi lado en los momentos buenos y los no tan buenos.

Índice general

Resumen v

Abstract vii

Índice general ix

Índice de figuras xiii

Índice de tablas xvii

Índice de algoritmos xix

Lista de acrónimos xxi

1. Introducción 1

1.1. Detección de comunidades . . . 1
1.2. Motivación . . . 3
1.3. Exposición del problema . . . 4
1.3.1. Preguntas de investigación . . . 5
1.4. Estructura . . . 6
1.5. Publicaciones . . . 6
1.5.1. Revistas científicas . . . 6
1.5.2. Congresos . . . 7

2. Estado del arte 9

2.1. Modelos de representación de redes . . . 9
2.1.1. Modelos estáticos . . . 9
2.1.2. Modelos dinámicos . . . 10
2.2. Detección de comunidades . . . 12
2.2.1. Detección de comunidades estáticas . . . 12

2.2.2. Detección de comunidades dinámicas 13
2.3. Medida o evaluación de la calidad de una comunidad 16
2.3.1. Evaluación de comunidades estáticas 17
2.3.2. Evaluación de comunidades dinámicas 18
2.4. Algoritmos genéticos en el ámbito de la detección de comunidades 20
2.4.1. Introdución a los algoritmos genéticos 20
2.4.2. Representación o codificación de GAs para la detección de comunidades . 20
2.4.3. Funciones de fitness en el ámbito de la detección de comunidades 22
2.4.4. MOGAs en el ámbito de la Detección de Comunidades 22
2.4.5. Algoritmos genéticos en problemas de carácter dinámico 26
2.4.5.1. Análisis a nivel poblacional 28
2.4.5.2. Análisis a nivel individual 29
2.4.6. Algoritmos genéticos para la detección de comunidades dinámicas 29

3. Técnicas eficientes para adaptar individuos a cambios en la red 31

3.1. Introducción 31
3.2. Diseño de los operadores 32
3.2.1. Reparación mediante label propagation 33
3.2.2. Reparación mediante random walks 35
3.2.3. Reparación híbrida 36
3.3. Caso de uso: CDP dinámicas en redes sintéticas y en Travian 36
3.3.1. Definición del problema 37
3.3.2. Diseño del algoritmo genético multi-objetivo 37
3.3.3. Batería de pruebas 40
3.3.4. Experimentación 44
3.3.4.1. Eficiencia de los operadores de reparación 44
3.3.4.2. Impacto de la reparación de poblaciones en el rendimiento del MOGA 46
3.3.4.3. Comparación con el estado del arte 52
3.4. Limitaciones y conclusiones 54

4. Metodologías eficientes para adaptar poblaciones a cambios en la red 57

4.1. Introducción 57
4.2. Metodología para la evaluación de estrategias de inmigrantes 59
4.2.1. Dimensiones a evaluar 59
4.2.2. Flujo de Trabajo 61
4.2.2.1. Paso uno: Comparación de los métodos bajo estudio con el estándar 61
4.2.2.2. Paso dos: Comparación de los métodos bajo estudio entre sí . . 62
4.2.2.3. Paso tres: Análisis de grano fino para identificar anomalías . . . 62
4.2.3. Técnica de visualización de la mezcla de poblaciones 63
4.3. Caso de uso: MOGA aplicados al problema de CDP dinámicas 64
4.3.1. Definición del problema 64
4.3.2. Estrategias bajo estudio 65
4.3.3. Instancias del CDP dinámico y configuración del MOGA 67
4.3.4. Aplicación de la metodología 69

4.3.4.1. Paso uno: Comparación de los métodos bajo estudio con el Estándar 69
4.3.4.2. Paso dos: Comparación de los métodos bajo estudio entre sí . . 70
4.3.4.3. Paso tres: Análisis de grano fino para identificar anomalías . . . 70
4.3.5. Conclusiones del caso de uso 77
4.4. Eficacia y limitaciones de la metodología 79

5. Conclusiones y trabajo futuro 81

5.1. Conclusiones 81
5.2. Trabajo Futuro 84

Apéndices 85

Bibliografía 203

Índice de figuras

2.1. Diagrama con modelos de redes complejas de diferente complejidad. Según se avanza de izquierda a derecha va aumentando la precisión del modelo, pero también su complejidad. Los modelos "clásico" y "agregado" no se pueden considerar modelos dinámicos, mientras que la "secuencia de instantáneas" y la "red temporal" sí. 10

2.2. Posibles eventos en el ciclo de vida de una comunidad [73, 18] 15

2.3. Taxonomia propuesta para clasificar los métodos de búsqueda de comunidades dinámicas. 16

2.4. Visualización de una estructura de comunidad dinámica. Cada fila corresponde a un nodo de la red y las columnas representan el tiempo de vida de la misma (cada columna es un instante temporal). El color de los nodos indica la comunidad a la que pertenecen, todos los nodos del mismo color pertenecen a la misma comunidad. Por último se marca un ejemplo de glitch y de pérdida de identidad. Imagen obtenida directamente del articulo [16]. 19

2.5. Diagrama de secuencia de un algoritmo génetico genérico. 21

2.6. Ejemplos de los diferentes tipos de codificaciones. La subfigura (a) muestra una red con 10 nodos y los colores las comunidades a las que perteneces. La subfigura (b) muestra un ejemplo de individuo que usa la codificación basada en etiquetas para codificar la estructura de comunidad de la subfigura (a); La subfigura (c) muestra un de individuo que usa la codificación basada en locus y debajo ver el resultado del paso intermedio de decodificación. Por último, la subfigura (d) muestra un individuo que usa la codificación basada en medoides. 23

2.7. Ejemplo de un frente de Pareto para un problema de maximización bi-objetivo genérico. Cada circulo representa una posible solución y los ejes X e Y el fitness de la solución en cada uno de los objetivos a optimizar. Las soluciones marcadas en rojo (A-H) corresponden al frente de Pareto y las marcadas en gris corresponden al área de soluciones dominadas. Nótese que para cada una de las soluciones marcadas con color gris es posible encontrar una solución en color rojo con mejor fitness en todos los objetivos. Imagen obtenida de wikimedia commons.[1] 24

2.8. Esquema del cálculo del hypervolumen para un problema de minimización bi-objetivo. 25

2.9. Comparación del Hipervolumen de dos conjuntos de soluciones no-dominadas. En azul las soluciones de un conjunto y en amarillo las del otro. Nótese que el color verde se utiliza para indicar soluciones que ambos conjuntos han encontrado. . . 26
2.10. Gráficas de los Paretos obtenidos por 10 ejecuciones de un mismo algoritmo para un problema de minimización bi-objetivo, donde los ejes X e Y representan los objetivos a minimizar. La subfigura (a) muestra los 10 paretos a la vez y la (b) las EAFs mejor, mediana y peor de los mismos resultados. En la subfigura (a), cada punto representa una de las soluciones no-dominadas encontradas por el algoritmo. Los resultados de cada ejecución se muestran con un color y una forma distinta. En la subfighura (b), la línea negra representa la peor EAF y la gris y la punteada la mediana y la mejor. 27

3.1. Ejemplo de las restricciones que impone la codificación basada en locus. Esta figura muestra en la derecha una red y en la izquierda los valores que pueden tener cada una de las posiciones del array de genes que conforma un individuo. Cualquier individuo que tenga un valor fuera de los permitidos en cualquier posición del array será considerado un individuo inválido. 32
3.2. Ejemplo de cómo un cambio en la red invalida un individuo. La imagen muestra una red temporal en dos instantes consecutivos T-1 y T. La arista punteada de color rojo en la red indica que dicha arista desapareció. Además de la red, se muestra un individuo que codifica una solución válida en el momento T-1 y como el mismo individuo ya no es válido en el momento T. 33
3.3. Ejemplo de reparación usando el método de *label propagation*. Las partes de la red con una línea continua son aquellas que participan en la reparación del nodo, mientras que las punteadas son ignoradas por el operador. El color del número de cada nodo indica la comunidad a la que pertenecía dicho nodo antes de que la red sufriera un cambio. 34
3.4. Ejemplo de un paseo aleatorio y del método de reparación basado en estos. Las partes de la red coloreadas son aquellas que han sido visitadas por un paseo aleatorio, mientras que las punteadas no. El color del número de cada nodo indica la comunidad a la que pertenecía dicho nodo antes de que la red sufriera un cambio. 36
3.5. Diagrama del flujo de ejecución del algoritmo evolutivo usado en el caso de uso. . 38
3.6. Diferencia de nodos y aristas entre dos instantáneas consecutivas de red de Mensajes en Travian. El eje Y muestra la diferencia de nodos/aristas, el X el día del mes en que se tomó la instantanea y el color de cada barra indica en número de nodos/aristas nuevo (verde), eliminado (rojo) o que permanecen (azul) respecto a la instantánea anterior. 41
3.7. Diferencia de nodos y aristas entre dos instantáneas consecutivas de red de Comercio en Travian. El eje Y muestra la diferencia de nodos/aristas, el X el día del mes en que se tomó la instantanea y el color de cada barra indica en número de nodos/aristas nuevo (verde), eliminado (rojo) o que permanecen (azul) respecto a la instantánea anterior. 41
3.8. Soluciones obtenidas para la red mx_03. La subfigura (a) contiene la diferencia de nodos entre dos instantáneas consecutivas y las subfiguras (b), (c) y (d) muestran el frente de Pareto obtenido para las instantáneas 2, 3 y 4 respectivamente. En los frentes de Pareto, los puntos verdes corresponden a los resultados del método basado en Label Propagation y los puntos rojos a los del método basado en Random Walks. 49

3.9. Arriba, se muestra la mediana del de generaciones utilizadas y, abajo, el tiempo de ejecución. La línea azul es para el método Estándar y la naranja para el método con reparación de poblaciones. Las líneas verticales indican los valores mínimo y máximo obtenidos en las 20 repeticiones y el punto el valor medio. 51

4.1. Diagrama de secuencia de un algoritmo genético con una técnica de inmigrantes. Los bloques de color claro indican los pasos del algoritmo genético clásico y los de color oscuro los bloques añadidos por la técnica de inmigrantes. 58
4.2. a) Flujo de trabajo para evaluar las dimensiones Calidad y Estabilidad. b) Flujo de trabajo para evaluar la dimensión Velocidad. 60
4.3. De izquierda a derecha, los resultados de la prueba de rango con signo de Friedman y el análisis post-hoc de Shaffer para las dimensiones de calidad, velocidad y estabilidad. Cada nodo representa un algoritmo diferente. Los nodos de color oscuro corresponden a las técnicas de inmigración en estudio y los claros a los demás métodos. El tamaño de los nodos es inversamente proporcional a su rango en la prueba de Friedman en la dimensión de Calidad y directamente proporcional en las de Velocidad y Estabilidad (es decir, los nodos más grandes tienen un mejor rendimiento). Dos nodos están conectados cuando la diferencia entre dos métodos no es estadísticamente significativa. 71
4.4. Las figuras de la izquierda (a, c, e, g) muestran las Attaiment Surfaces de los últimos frentes de Pareto encontrado por el Estándar (panel izquierdo) y la estrategia inmigrantes (panel derecho). Las figuras de la derecha (b, d, f, h) muestran la evolución del HV a lo largo de las diferentes generaciones del AGMO para los mismos métodos. El Estándar se muestra en naranja, mientras que la estrategia del inmigrante se muestra en azul. 72
4.5. Número de individuos *mestizos* para el método Random Immigrant Genetic Algoritm (Algoritmo genético con inmigrantes aleatorios) (RIGA) que sobreviven (en naranja) y son expulsados (en azul) de la población entre dos generaciones consecutivas. El eje X muestra el número de generaciones y el eje Y el número de individuos de un tipo determinado. El porcentaje en la esquina inferior derecha del gráfico indica el porcentaje de inmigrantes insertados en la población en cada generación. 74
4.6. Número de individuos de cada tipo para el método HIGA. El eje X muestra el número de generaciones y el eje Y el número de individuos de un tipo determinado. La subfigura (a) muestra el número de individuos que sobreviven de una generación a otra y la subfigura (b) muestra el que son expulsados. 75
4.7. Número de individuos de cada tipo para el método EIGA. El eje X muestra el número de generaciones que dura el proceso evolutivo y el eje Y el número de individuos de un tipo concreto. El porcentaje en la esquina superior derecha del gráfico indica el porcentaje de inmigrantes insertados en la población en cada generación. 76
4.8. Número de individuos de cada tipo para el método DEIGA. El eje X muestra el número de generaciones durante el proceso evolutivo, mientras que el eje Y muestra el número de individuos de un tipo concreto. El número en la esquina superior derecha del gráfico indica el número de generaciones esperadas hasta que los inmigrantes se insertan en la población. En todos los gráficos se inserta un 10 % de inmigrantes en cada generación. 78

Índice de tablas

3.1. Nomenclatura seguida en las redes dinámicas sintéticas con un número fijo de nodos. 42
3.2. Nomenclatura seguida en las redes dinámicas sintéticas con un número variable de nodos. 43
3.3. parámetros del MOGA 45
3.4. Hipervolumen obtenido por los operadores de reparación para redes sintéticas con un número fijo de nodos. Un tick indica que hay suficiente evidencia estadística para considerar los resultados los diferentes, mientras que una cruz indica lo contrario. Cada celda muestra el valor medio y la desviación estándar de las 20 repeticiones. 47
3.5. Hipervolumen obtenido por los operadores de reparación para redes sintéticas con un número variable de nodos. Un tick indica que hay suficiente evidencia estadística para considerar los resultados los diferentes, mientras que una cruz indica lo contrario. Cada celda muestra el valor medio y la desviación estándar de las 20 repeticiones. 48
3.6. Error cuadrático medio (1-NMI). Cada celda muestra el valor medio y la desviación estándar de 20 repeticiones. 50
3.7. Error cuadrático medio (1-NMI). Cada celda muestra el valor medio y la desviación estándar de las 20 repeticiones. Una celda marcada con "$nan \pm nan$" indica que un algoritmo no fue capaz de procesar la red correspondiente. Por último, un tick (✓) indica que hay suficiente evidencia estadística para considerar distinto el rendimiento entre los algoritmos, mientras que una cruz (✗) indicaría lo contrario 53
3.8. NMI obtenidos en cada instantánea de la red de mensajes de Travian. Cada celda muestra el valor medio y la desviación estándar de las 20 repeticiones. Un tick (✓) indica que hay suficiente evidencia estadística para considerar distinto el rendimiento entre los algoritmos y una cruz (✗) indica lo contrario 55
3.9. NMI obtenidos en cada instantánea de la red de comercio de Travian. Cada celda muestra el valor medio y la desviación estándar de las 20 repeticiones. Un tick (✓) indica que hay suficiente evidencia estadística para considerar distinto el rendimiento entre los algoritmos y una cruz (✗) indica lo contrario 56

4.1. Las filas y columnas muestran el tipo de los padres, mientras que las celdas representan el tipo de su progenie. Como la tabla es simétrica, sólo se muestra la diagonal superior de la tabla. 64

4.2. Descripción de las 18 redes dinámicas seleccionados para el caso de uso. 68
4.3. Configuración del MOGA del caso de estudio 68
4.4. Resultados de la prueba de rango con signo de Friedman con análisis post-hoc de Holm utilizando el Estándar como método de control. El primer grupo contiene las tres estrategias de inmigrantes, y el segundo el resto de los métodos. ▲ indica que un método tiene un mejor rendimiento que el Estándar, por el contrario, ▼ indica que un método funciona peor y, por último, ≈ indica que el método tiene el mismo rendimiento. 69
4.5. Resultados de la prueba de rango con signo de Friedman y del análisis post-hoc de Holm sobre el HV de las soluciones y usando el Estándar como método de control. 73

Índice de Algoritmos

1. Reparar población 33
2. Reparación mediante label propagation 34
3. Reparación mediante random walks 35
4. Reparación mediante la técnica híbrida 37
5. Proceso evolutivo 39
6. mutacion 39
7. RIGA 65
8. EIGA 66
9. HIGA 67

Lista de acrónimos

CDP Community Detection Problem (Problema de detección de comunidades) 1
SNA Social Network Analysis (Análisis de redes complejas) 3
NMI Normalized Mutual Information 18
GA Genetic Algorithm (Algoritmos genéticos) 4
MOGA Multi-Objective Genetic Algorithm (Algoritmos genéticos multi-objetivo) 20
EAF Emperical Attaiment Function 25
HV Hipervolumen 25
RIGA Random Immigrant Genetic Algoritm (Algoritmo genético con inmigrantes aleatorios) XV
EIGA Elite Immigrant Genetic Algoritm (Algoritmo genético con inmigrantes elitistas) 28
HIGA Hybrid Immigrant Genetic Algoritm (Algoritmo genético con inmigrantes híbridos) 28
PSO Particle Swarm Optimization (Optimización basada en nubes de partículas) . . . 29
HGA Hypermutation Genetic Algorithm (Algoritmo genético de hipermutación) 66
DEIGA Delayed Elite Immigrants Genetic Algorithms (Algoritmo genético con inmigrantes elitistas desfasado) 66
ROOT Robust Optimization Over Time (Optimización robusta duradera en el tiempo) . 13
EMTO Evolutionary Multi-Task Optimisation (Optimización evolutiva multitarea) . . . 79

Capítulo 1

Introducción

Este capítulo presenta una visión general de los temas abordados en esta tesis, así como sus principales aportaciones y las motivaciones de la misma. En él se hace una breve introducción al problema de detección de comunidades dinámicas y su contexto. La intención de este capítulo es introducir al lector de una manera progresiva en la temática de esta tesis, sin entrar en detalles técnicos, ya que en su capitulo 2 se hará una exposición precisa del estado del arte asi como una definición formal del problema.

El capítulo comienza haciendo un breve resumen del problema de la detección de comunidades tanto en su versión estática como dinámica (sección 1.1); una vez definido el problema, se formularán las motivaciones de esta tesis (sección 1.2); a continuación se expondrán las principales aportaciones de la misma (sección 1.3); por último, se enumerará la estructura de este documento (sección 1.4) y las publicaciones generadas durante la realización de esta tesis (sección 1.5).

1.1. Detección de comunidades

Dado un grafo o red compuesto por nodos y aristas, el Community Detection Problem (Problema de detección de comunidades) (CDP) consiste en dividir dicho grafo en grupos de nodos, a partir de ahora conocidos como "comunidades", teniendo en cuenta la topología de la red. En pocas palabras, podría considerarse como un proceso de *clustering* sobre grafos. No existe una definición única de Comunidad acordada por la comunidad científica, pero entre esta se considera que una comunidad es de calidad cuando presenta más conexiones internas que externas, es decir, que los nodos de la comunidad tienen más aristas entre los otros miembros de la comunidad en comparación con el resto de los nodos del grafo [105].

En las últimas décadas, el CDP ha recibido mucha atención por parte de la comunidad investigadora. Sin embargo, la mayoría de los trabajos publicados hasta la fecha se centran en trabajar con grafos estáticos, es decir, que no cambian en el tiempo. A este tipo de algoritmos y las comunidades que detectan se les denomina *estáticas* [36]. Está limitación es poco realista para muchos problemas del mundo real y en los últimos años la comunidad científica ha comenzado a centrar sus esfuerzos en desarrollar algoritmos que sean capaces de trabajar con grafos dinámicos, es decir, que cambian en el tiempo [85].

Independientemente de que los grafos sean estáticos o dinámicos el CDP se puede abordar desde múltiples ángulos. A lo largo de los años se han propuesto decenas de algoritmos eficientes para encontrar comunidades de todo tipo en cualquier tipo de grafo [72]. Ciertos algoritmos se centran en encontrar jerarquías de comunidades mientras que otros se centran en buscar una única división de comunidades. De la misma manera, ciertos algoritmos fuerzan a que cada nodo solo pertenezca a un único grupo, dando lugar a "comunidades nítidas" o "no-solapadas", mientras que otros métodos no imponen esta restricción y permiten que un mismo nodo pertenezca a varios grupos a la vez, dando lugar a "comunidades difusas" o "solapadas.

El permitir que los grafos evolucionen en el tiempo amplía la cantidad de análisis que podemos hacer sobre una red. Cuando tenemos en cuenta el tiempo, ya no sólo podemos analizar las comunidades en cada instante temporal, sino que, también, nos puede interesar buscar momentos en el tiempo en el que las comunidades cambian de forma brusca; tanto identificar como predecir el ciclo de vida de una comunidad (cuando aparece, si crece o decrece, si se divide en varias comunidades... etc.); o buscar estructuras estables en el tiempo, en casos dónde las redes sean muy ruidosas; entre otras muchas posibilidadess. Todos estos análisis, tan diferentes entre sí, están englobados dentro del paraguas de la detección dinámica de comunidades [85].

Incluir la variable o dimensión **temporal** en una red no es trivial y para ello es necesario expandir el modelo clásico de grafo. Cambiar el modelo de grafo acarrea un problema ya que cuanto más nos alejemos del grafo clásico menos compatibles son los métodos estáticos de detección de comunidades, pero es necesario sí se pretende tener más precisión temporal. Por esta razón se han desarrollado varias extensiones del modelo clásico de grafo de precisión y complejidad incremental, entre las que caben destacar la *secuencia de instantáneas* y la *red temporal* [85]. La *secuencia de instantáneas* consiste en plasmar el estado de la red en varios momentos a lo largo de su vida, como si de una secuencia de fotos o instantáneas se tratase. Por el contrario, la *red temporal* evita hacer agregación alguna y modela la red como un conjunto de aristas etiquetadas con su tiempo de inicio y su duración.

La elección de uno de los modelos anteriores frente a otro depende mucho del problema que se vaya a tratar y del análisis que se pretenda hacer. Por ejemplo, para usar el modelo de *secuencia de instantáneas* es necesario definir la frecuencia con la que se toman las instantáneas. Este proceso puede ser trivial para problemas estacionales o en los que exista una rutina implícita, como el estudio de la movilidad de la gente en una ciudad, sin embargo, puede ser todo un desafío en problemas dónde dicha rutina no exista, por ejemplo, analizando conversaciones cara a cara durante un evento. Por otro lado, en la *red temporal*, cada arista debe tener un tiempo de vida acotado. Definir dicho tiempo de vida es banal en problemas dónde las interacciones tengan una duración conocida, como las llamadas telefónicas, sin embargo, puede llegar a ser un quebradero de cabeza en problemas dónde dicha duración no se conozca, por ejemplo, comunicaciones por email.

De la misma forma, según el tipo de análisis que queramos realizar habrá un modelo más idóneo que otro. Por ejemplo, si queremos buscar momentos en el tiempo dónde la red sufre cambios bruscos, el modelo más idóneo es de red temporal ya que en la secuencia de instantáneas la frecuencia de toma de instantáneas sesga el análisis. Por el contrario, el modelo de la secuencia de instantáneas nos simplifica la vida a la hora de identificar o predecir el ciclo de vida de las comunidades, ya que es mucho más sencillo detectar diferencias analizando la misma comunidad en dos instantes concretos que encontrar el momento exacto en el que una comunidad sufre un cambio lo suficientemente grande para considerarse de interés.

1.2. Motivación

El Social Network Analysis (Análisis de redes complejas) (SNA) es un área de investigación en auge dada la cantidad de problemas del mundo real que se prestan a ser modelados como una red [13] . Problemas tan diversos como pueden ser la interacción entre proteínas o la movilidad en transporte público en una ciudad se pueden beneficiar de los avances de SNA. El campo de SNA se compone de diferentes tipos de análisis como el de la centralidad (buscar qué nodos son más importantes dentro de la red), o el de la predicción de links (buscar qué aristas son más probables que ocurran) [105]. De entre todos los análisis incluidos en el campo de SNA, destaca el CDP sobre todo tras el auge de las redes sociales e internet ya que nos permite sintetizar conocimiento sobre redes enormes.

Como ya indicamos anteriormente, el CDP ha recibido mucha atención por parte de la comunidad científica en las últimas décadas, sobre todo en la versión estática del problema. Como en tantas otras doctrinas, sólo desde hace poco que se ha comenzado a prestar atención a las versiones dinámicas de los problemas y el CDP no es una excepción. La versión dinámica del CDP está aun inmadura comparada con la versión estática y tiene varios desafíos por delante.

1. **Propiedades que debe tener una comunidad dinámica**: El primero desafio indentificado consiste en definir qué propiedades deben de tener las comunidades dinámicas. Es obvio que las comunidades deben de tener más conexiones internas que externas, pero además de esta propiedad también debe existir una continuidad en el tiempo de dichas comunidades. Un método que sólo encuentra comunidades esporádicas, que sólo existen durante periodos ínfimos de tiempo, y que, por lo tanto, es imposible obtener ninguna información sobre sus dinámicas, no es muy valioso. El CDP dinámico se encuentra en un estado temprano en este aspecto. La mayoría de algoritmos que encontramos en el estado del arte se comparan solo midiendo la "calidad" de las comunidades, es decir cómo de interconectadas están las comunidades que encuentran, obviando la estabilidad en el tiempo de las mismas. No ha sido hasta hace poco que en el estado del arte han comenzado a aparecer publicaciones abordando este tema y proponiendo métricas para evaluar la calidad de las comunidades dinámicas de una manera más realista [16].

2. **Algoritmos capaces de encontrar comunidades dinámicas de calidad**: El segundo desafío que tiene por delante el área, es diseñar algoritmos que encuentren comunidades con las propiedades descritas en el desafío anterior. Es curioso y sorprendente cómo se desarrollan las áreas de investigación puesto que el CDP dinámico está mucho más avanzado en este desafío comparado con el anterior. Por ejemplo, Chakrabarti et al. [19] propusieron un framework llamado "evolutionary clustering" con el que tratan de abordar la problemática del punto anterior. El "evolutionary clustering" define dos fuerzas contrapuestas a la hora de buscar comunidades estáticas. Una fuerza mueve las comunidades a que se ajusten lo mejor posible a la topología de una red en un momento dado del tiempo, mientras que la otra fuerza constriñe a las comunidades para que no se alejen demasiado de su historia reciente. Se han propuestos algoritmos de toda índole para encontrar comunidades usando "evolutionary clustering". Sin embargo, el problema de equilibrar la importancia de cada una de las dos fuerzas aún no está resulto y por ahora es necesario hacerlo de manera manual para cada problema a base de ensayo y error.

3. **Algoritmos eficientes aplicables en entornos reales**: El tercer desafío que se ha iden-

tificado, consiste en desarrollar métodos eficientes en tiempo y memoria para encontrar comunidades dinámicas. Permitir que las redes evolucionen en el tiempo implica un aumento considerable en la cantidad de cálculo necesario para analizarlas. Es de vital importancia desarrollar algoritmos que consuman una cantidad de recursos (memoria, tiempo, necesidades de cálculo ... etc.) tolerables si queremos aplicarlos a problemas del mundo real. Al igual que en el desafío anterior, la comunidad científica ya ha comenzado a movilizarse y ha propuesto maneras ingeniosas y eficientes de calcular algunos de los métodos clásicos desarrollados para la versión estática del CDP [85]. Aun así, este desafío está lejos de ser resuelto y se necesita más trabajo para tener métodos capaces de trabajar en entornos reales consumiendo una cantidad de recursos asumibles.

4. **Categorización y sistematización del área**: El cuarto y último desafío identificado, está relacionado con la heterogeneidad del área. Como ya hemos comentado en el desafío anterior, al incluir la variable tiempo la cantidad de análisis posibles se incrementa. Dentro del paraguas del CDP dinámico se encuentran muchos algoritmos cuyas entradas y salidas son muy dispares y que, a menudo, solo resuelven alguna parte del problema [85]. Tenemos algoritmos que dada una red temporal devuelve una única separación en comunidades; otros que, por el contrario, nos devuelven varias separaciones en comunidades a lo largo del tiempo; otros que dada una secuencia de separaciones a lo largo del tiempo calculan el ciclo de vida de una comunidad; otros que no devuelven comunidades per se, sino que sólo devuelven momentos dónde la red sufre cambios bruscos ... etc. Es necesario imponer ordenar y agrupar dichos métodos de una manera coherente para que un neófito sea capaz de decidir qué análisis son posibles y qué métodos tiene disponibles para llevarlos a cabo. Aunque podemos encontrar varios trabajos en el estado del arte dónde se categorizan los algoritmos propuestos [85, 26, 17], considero que el desafío sigue sin estar resuelto y que, aunque son un primer paso para ellos, son necesarios trabajos más amplios que estudien el problema desde el punto de vista de las entradas y salidas de los métodos y no solo teniendo en cuenta el tipo de algoritmo que se utiliza. De la misma manera, esta "organización" también debe verse reflejada en el área del software y los flujos de trabajo. A día de hoy no hay una librería, framework o software lo suficientemente maduro que aglutine flujos de trabajo completos para detectar comunidades dinámicas. Hay algunos intentos tempranos[1], pero no son comparables al software disponible para el CDP estático[2].

1.3. Exposición del problema

Esta tesis se centra en el tercer desafío identificado en la sección anterior, que puede resumirse en el **desarrollo de algoritmos eficientes aplicables en entornos reales**. Específicamente, en el desarrollo de Genetic Algorithm (Algoritmos genéticos) (GA) eficientes para la detección de comunidades dinámicas. Los GAs son una de las familias más populares que forman parte del conjunto de algoritmos meta-heurísticos bio-inspirados. Dichos algoritmos han sido aplicados en centenares de dominios de forma satisfactoria, incluido el CDP [81]. Las meta-heurísticas bio-inspiradas ofrecen un compromiso excelente entre potencia, flexibilidad y escalabilidad. Su mejor ventaja es que son capaces de generar soluciones con una calidad aceptable en un tiempo de ejecución acotado, cosa que los métodos clásicos no pueden ofrecer. Cuando un problema crece

[1] Tnetwork (`https://tnetwork.readthedocs.io/en/latest/`)
[2] NetworkX (`https://networkx.org/`)

hasta tamaños dónde los métodos clásicos dejan de ser aplicables, ya sea por tiempo o memoria, un GA (o en su defecto cualquier meta-heurística bio-inspirada) es un candidato excelente para obtener soluciones con una calidad aceptable, consumiendo una cantidad de recursos asumible.

Aplicar un GA a un entorno dinámico de manera satisfactoria no es trivial y es necesario definir mecanismos extra. Los GAs son heurísticos poblacionales, es decir, durante el proceso de ejecución no existe una única solución que se va mejorando, sino que existe un conjunto de ellas. Cuando trabajamos en un entorno dinámico y aparecen cambios, es posible que la población deje de ser capaz de encontrar soluciones satisfactorias al problema. Una solución trivial a este problema es descartar la población actual cuando aparezca un cambio y crear una nueva completamente de cero. Sin embargo, dicha estrategia descarta todo trabajo pasado, lo que la convierte en ineficiente y no aplicable al mundo real. Por esa razón, si queremos GA capaces de ser aplicados en entornos dinámicos del mundo real es necesario transformar las poblaciones en dos niveles distintos:

- **A nivel individual**: Dependiendo del problema, es posible que cambios en el entorno ocasionen que individuos que codificaban soluciones completamente válidas al problema degeneren y pasen a codificar soluciones inválidas, es decir, soluciones imposibles o que incumplen alguna restricción. Para los problemas dónde esto pueda suceder, es necesario definir *operadores de reparación* que sean capaces de arreglar estos individuos inválidos para que codifiquen soluciones válidas de nuevo. Dichos reparadores dependen en gran medida del problema a tratar y de la codificación elegida, por lo tanto, no hay recetas generales que se puedan usar y se deben desarrollar métodos concretos para cada problema.
- **A nivel poblacional**: Durante el proceso evolutivo la población de un GA se va especializando para encontrar soluciones prometedoras al problema en cuestión. Cuando aparece un cambio en el entorno y modifica el problema, es posible que la población actual esté tan ajustada al entorno anterior al cambio que sea imposible para ella seguir buscando soluciones prometedoras en el entorno nuevo. Por esta razón, en cualquier problema dinámico, es necesario definir mecanismos que permitan adaptar las poblaciones a los cambios del entorno y asi evitar que estas pierdan capacidades. Dado que este fenómeno es común a cualquier problema, existen varias técnicas como la hipermutación, las estrategias de memoria o las estrategias de inmigrantes, entre otras, que se pueden usar para adaptar la población de un GA a un cambio [71]. Sin embargo, qué estrategia es la más adecuada para cada caso es aún un problema abierto.

1.3.1. Preguntas de investigación

Teniendo en mente lo anterior, esta tesis se propone responder a las siguientes preguntas de investigación (RQ o Research Questions):

- **RQ1**: ¿Cuál es el método más eficiente para adaptar los individuos de un GA diseñado para la detección de comunidades a los cambios del entorno?
- **RQ2**: Entre las estrategias presentes en el estado del arte para adaptar la población de un GA a cambios en el entorno ¿Cuál de ellas funciona mejor para la versión dinámica del CDP y qué ventajas e inconvenientes presenta?

- **RQ3**: Del conocimiento obtenido durante la realización de esta tesis ¿cuál es generalizable a otros dominios?, ¿Cómo puede elegirse la mejor estrategia para adaptar las poblaciones de un GA en problemas ajenos al CDP dinámico?

1.4. Estructura

Este documento se ha estructurado en cinco capítulos que se resumirán a continuación:

- **Capítulo 1 - Introducción**: Capítulo dónde se realiza una breve introducción a la temática de esta tesis, se exponen sus motivaciones y las principales aportaciones de la misma.
- **Capítulo 2 - Estado del arte**: En este capítulo se formula el problema de la detección dinámica de comunidades, se hace una revisión del estado del arte del mismo, además de una revisión exhaustiva sobre la aplicación de GAs para la detección dinámica de comunidades.
- **Capítulo 3 – Técnicas eficientes para adaptar individuos a cambios en el entorno**: Este capítulo propone varios operadores de reparación capaces de arreglar individuos inválidos en GAs para detectar comunidades dinámicas. Además, se valida su funcionamiento aplicándolos en un GA concreto y se mide su eficiencia con redes tanto sintéticas como reales. Este capítulo responde la pregunta de investigación RQ1.
- **Capítulo 4 - Metodologías eficientes para adaptar poblaciones a cambios del entorno**: De las posibles técnicas del estado del arte para adaptar poblaciones a los entornos dinámicos, esta tesís trabaja con una de las más populares, las estrategias de inmigrantes. En este capítulo se propone una metodología genérica para evaluar estrategias de inmigrante en problemas dinámicos y se aplica al problema de la detección dinámica de comunidades. Este capítulo responde las preguntas de investigación RQ2 y RQ3.
- **Capítulo 5 – Conclusiones y trabajo futuro**: Se responden a las preguntas de investigación planteadas en el capítulo 1 y se proponen futuras líneas de investigación.

1.5. Publicaciones

A continuación, se enumeran las publicaciones generadas durante esta tesis y las aportaciones de cada publicación. Las publicaciones se han divido entre revistas científicas y congresos. Todas las publicaciones están disponibles, usando un formato no propietario, en los anexos de este documento. Dentro de cada sección las publicaciones se han ordenado por fecha de publicación.

1.5.1. Revistas científicas

- **R1**: David Camacho, **Ángel Panizo-LLedot**, Gema Bello-Orgaz, Antonio Gonzalez-Pardo, and Erik Cambria (2020). The four dimensions of social network analysis: An overview of research methods, applications, and software tools. Information Fusion, 63,

88-120.
Índice de impacto: 12,97 (JCR Q1/D1)

- Este artículo ha contribuido al capítulo 2 de este documento, y han ayudado a determinar algunas de las técnicas y metodologías que se han desarrollado en los capítulos 3 y 4.
- Una versión de este documento está disponible, sin el formato de la revista, en los anexos de este tomo.

- **R2**: **Ángel Panizo-LLedot**, Gema Bello-Orgaz, and David Camacho. (2020). A multi-objective genetic algorithm for detecting dynamic communities using a local search driven immigrant's scheme. Future Generation Computer Systems, 110, 960-975.
  Índice de impacto: 7,187 (JCR Q1)

  - Este artículo ha contribuido al capítulo 3 de este documento.
  - Una versión de este documento está disponible, sin el formato de la revista, en los anexos de este tomo.

- **R3**: **Ángel Panizo-LLedot**, Martín Pedemonte, Gema Bello-Orgaz and David Camacho, Addressing evolutionary-based dynamic problems: a new methodology for evaluating Immigrants strategies in MOGAs in IEEE Access, doi: 10.1109/ACCESS.2022.3156944.
  Índice de impacto: 3,367 (JCR Q2)

  - Este artículo ha contribuido al capítulo 4 de este documento.
  - Una versión de este documento está disponible, sin el formato de la revista en los anexos de este tomo.

1.5.2. Congresos

- **C1**: **Ángel Panizo-LLedot**, Gema Bello-Orgaz, and David Camacho (2018, October). A genetic algorithm with local search based on label propagation for detecting dynamic communities. In International symposium on intelligent and distributed computing (pp. 319-328). Springer, Cham.

  - Este artículo ha contribuido al capítulo 3 de este documento.
  - Una versión en abierto de este articulo está disponible en ResearchGate[3]

- **C2**: **Ángel Panizo-LLedot**, Gema Bello-Orgaz, Alfonso Ortega , and David Camacho (2018). Un Algoritmo Memético, con búsqueda local basada en Label Propagation, para detectar comunidades en redes dinámicas. In XVIII Conferencia de la Asociación Española para la Inteligencia Artificial (CAEPIA 2018) 23-26 de octubre de 2018 Granada, España (pp. 995-1000). Asociación Española para la Inteligencia Artificial (AEPIA).

  - Este artículo ha contribuido al capítulo 3 de este documento.
  - Una version en abierto de este documento, sin el formato de la revista, está disponible en ResearchGate[4]

[3] https://www.researchgate.net/publication/326712986_Community_finding_in_dynamic_networks_using_a_genetic_algorithm_improved_via_a_hybrid_immigrants_scheme

[4] https://www.researchgate.net/publication/357888845_UN_ALGORITMO_MEMETICO_CON_BUSQUEDA_LOCAL_BASADA_EN_LABEL_PROPAGATION_PARA_DETECTAR_COMUNIDADES_EN_REDES_DINAMICAS

- **C3**: **Ángel Panizo-LLedot**, Gema Bello-Orgaz, Alfonso Ortega, and David Camacho (2018). Community finding in dynamic networks using a genetic algorithm improved via a hybrid immigrants scheme. In Data Science and Knowledge Engineering for Sensing Decision Support: Proceedings of the 13th International FLINS Conference (FLINS 2018) (pp. 591-598).
  - Este artículo ha contribuido al capítulo 3 de este documento.
  - Una version en abierto de este documento, sin el formato de la revista, está disponible en ResearchGate: [5]

[5] `https://www.researchgate.net/publication/326712986_Community_finding_in_dynamic_networks_using_a_genetic_algorithm_improved_via_a_hybrid_immigrants_scheme`

Capítulo 2

Estado del arte

En este capítulo se realiza una revisión exhaustiva del estado del arte sobre el CDP dinámico, así como una revisión general del estado del arte en el problema de la Detección de Comunidades. Se comenzará enumerando las principales formas de modelar una red dinámica y los desafíos que surgen de usar cada una (sección 2.1). A continuación, se hará un repaso de los posibles análisis amparados bajo el paraguas de la detección de comunidades (sección 2.2) y las estrategias disponibles para evaluar su calidad (sección 2.3). Por último, se expondrán las principales técnicas usadas para aplicar GA para la detección de comunidades en general, haciendo especial énfasis en las técnicas aplicadas para la versión dinámica del problema (sección 2.4).

2.1. Modelos de representación de redes

En esta sección se definen formalmente los modelo matemáticos diseñados para representar redes. Primero se plantea una definición formal de grafo y se presentan algunas alternativas. A continuación, se analiza en profundidad las implicaciones que tiene introducir la varibale "tiempo" en el modelo de grafo y se definen formalmente los dos modelos de redes dinámicas más utilizados.

2.1.1. Modelos estáticos

La forma por antonomasia de representar una red es el *grafo*. Un grafo $G = (V, E)$ es un modelo matemático que permite representar entidades y relaciones entre ellas. Está compuesto por un conjunto $V = \{v_1, .., v_n\}$ de *nodos* o *vertices* y otro conjunto $E = \{e_{ij} | v_i \in V \wedge v_j \in V\}$ de *aristas* o *enlaces* entre nodos.

La manera más común de representar un grafo es mediante su *matrix de adyacencia* (A), que se define como una matriz cuadrada de n x n, donde n indica el número de nodos en el grafo y cada coeficiente a_{ij} satisface la Ecuación 2.1.

$$a_{ij} = \begin{cases} k, if\ e_{ij} \in E \\ 0, otherwise \end{cases} \tag{2.1}$$

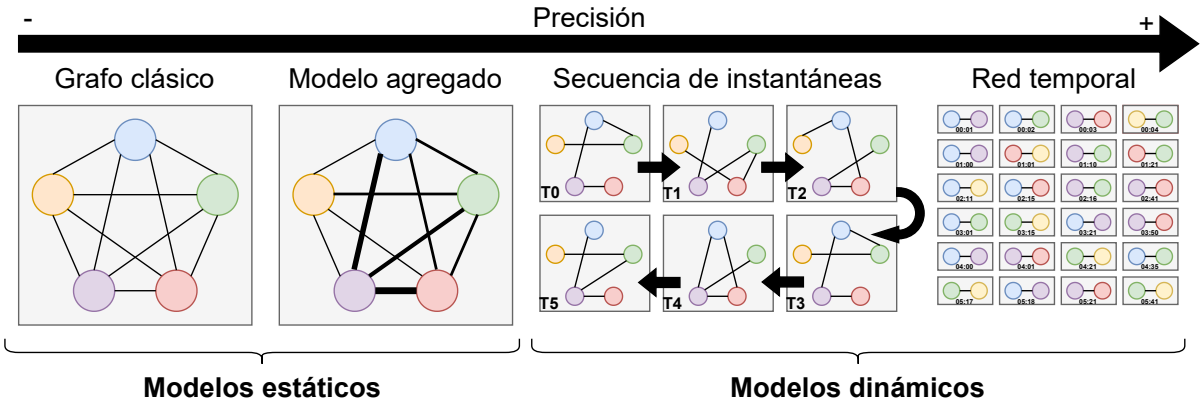


Figura 2.1: *Diagrama con modelos de redes complejas de diferente complejidad. Según se avanza de izquierda a derecha va aumentando la precisión del modelo, pero también su complejidad. Los modelos "clásico" y "agregado" no se pueden considerar modelos dinámicos, mientras que la "secuencia de instantáneas" y la "red temporal" sí.*

En el *modelo clásico de grafo* el valor k siempre tiene que ser igual a 1. Sin embargo, cuando se permite que el valor de k tome valores distintos de 1 entonces el grafo se pasa a denominar *grafo ponderado*. Por último, un grafo se considera *dirigido* si $\exists i, j, a_{ij} \neq a_{ji}$ y *no-dirigido* en caso contrario, es decir, $\forall i, j, a_{ij} = a_{ji}$.

2.1.2. Modelos dinámicos

Incluir la característica (o variable) temporal en el modelo clásico de grafo no es trivial. Por ello, Rossetti y Cazabet presentaron en [85] varios modelos de creciente complejidad y precisión, tal como muestra el diagrama de la Figura 2.1, donde la precisión se define como la capacidad del modelo de indentificar el momento exacto en el que ocurre una interacción así como su duración. De izquierda a derecha, el diagrama muestra cada vez modelos más precisos, pero a su vez más complejos:

- El primero de ellos es el modelo clásico de grafo mencionado anteriormente. Este modelo tiene una precisión temporal nula, puesto que es incapaz de capturar ninguna información temporal. Nótese, que este modelo almacena las interaciones entre las distintas entidades de la red, pero es imposible saber cuánto duraron, cuándo ocurrieron o con qué frecuencia.
- El siguiente modelo en nivel de complejidad es el modelo *agregado*. Este modelo consiste en generar un grafo ponderado, donde los pesos de las aristas representan el número de veces que se ha producido una interacción. El modelo *agregado*, al igual que el clásico, no permite saber cuando ocurrió una interacción o su duración. Sin embargo, al contrario que el modelo clásico, si permite saber con que frecuencia ocurrió, lo que posibilita realizar algún tipo de análisis dinámico, como la estimación de la fuerza de una arista. Aun así, estos dos primeros modelos, el *grafo clásico* y el *modelo agregado*, no se consideran modelos dinámicos.
- El primer modelo en nivel de complejidad que podemos considerar dinámico, es el de la *secuencia de instantáneas* (o *snapshots* del inglés). Este modelo consiste en plasmar el

estado de la red en varios momentos a lo largo de su vida, como si de una secuencia de fotos o instantáneas se tratase. Este es el primero modelo que permite saber cuando ocurrio una interación concreta y su duración, aunque dicho valor sea una orquilla de tiempos y no un valor concreto. En la práctica, dicho modelo se representa como una secuencia de grafos $SN = \{G^1, ..., G^T\}$ donde cada G^t representa la red en un momento concreto del tiempo. Este modelo muestra un compromiso entre los modelos clásicos de grafos y la precisión temporal.

- El último modelo y el más preciso (en cuanto a la precisión con la que se indentifica una interacción) de todos es la *red temporal*. En una red temporal $TN = (V, E^T)$ el conjunto de aristas $E^T = \{(e_{ij}, t, \delta)|v_i \in V \wedge v_j \in V\}$ se marcan con un tiempo de inicio (t) y una duración (δ). Sin lugar a dudas, la red temporal es el modelo más preciso de todos, ya que nos permite saber el momento exacto en el que ocurre una interacción y su duración pero también el más complejo y el que más se aleja del modelo clásico de grafo haciéndolo el menos compatible con los métodos clásicos.

Usar cualquiera de los dos modelos anteriores, la secuencia de instantáneas o la red temporal, implica tomar una serie de decisiones que pueden tener un impacto acusado en los resultados [14, 85]. Por ejemplo, al usar la secuencia de instantáneas, es necesario definir la granularidad con la que se tomarán las mismas. Si se elije una granularidad muy grande es posible que pierda eventos, mientras que, si elijo una muy pequeña puedo generar redes en apariencia estables cuando en verdad no lo son. Por otro lado, al usar la red temporal es necesario definir el tiempo de inicio y la duración de todas las aristas. Esto puede ser problemático en situaciones donde cualquiera de estos dos valores no esté claro. Por ejemplo, al analizar interacciones por email dentro de una empresa, el tiempo de inicio está claro pero no así su duración. Dependiendo del valor que le demos, los resultados pueden verse drásticamente afectados. Este fenómeno, hace que la elección de un modelo frente a otro dependa mucho del tipo de problema sobre el que se trabaje y de el análisis que se quiera realizar. Por ejemplo, en problemas donde existe una rutina o estacionalidad implícita (véanse los emails dentro de una empresa), es más conveniente utilizar la secuencia de instantáneas. Tengáse en cuenta que para usar la red temporal en dicho problema se tiene que definir una duración estándar (un día, una semana, un mes... etc) y fija para cada interación de la red (email). Este hecho implica inbuir de cierta granularidad al modelo de red temporal lo que introduce sesgos en el modelo que reducen su precisión. Puesto que no se puede aprovechar toda la precisón del modelo de red temporal, en este caso, es mejor usar el modelo de la secuencia de instantaneas que es más sencillo. Por el contrario, en problemas donde las interacciones no sigan rutina alguna y tengan una duración bien definida (como por ejemplo, las interacciones personales espontáneas que se originan en una fiesta o en cualquier evento público), es más conveniente utilizar el modelo de red temporal. Dada la naturaleza del susodicho problema, si se usara el modelo de la secuencia de instántaneas, encontrar la frecuencia correcta con la que tomarlas sería muy complicado y un problema en sí mismo. Dado el caso, sería mejor no hacer agregación alguna y decantarse por el modelo de red temporal.

De la misma forma, según el tipo de análisis que queramos realizar habrá un modelo más idóneo que otro. Por ejemplo, si queremos buscar momentos en el tiempo donde la red sufre cambios bruscos, el modelo más idóneo es el de red temporal ya que en la secuencia de instantáneas la frecuencia de toma de instantáneas sesga el análisis. Por el contrario, el modelo de la secuencia de instantáneas nos simplifica la vida a la hora de identificar o predecir el ciclo de vida de las comunidades, ya que es mucho más sencillo detectar diferencias analizando la misma comunidad

en dos instantes concretos que encontrar el momento exacto en el que una comunidad sufre un cambio lo suficientemente grande para considerarse de interés.

Además de las consideraciones de los párrafos anteriores, usar estos modelos también conlleva una serie de desafíos. Al usar la secuencia de instantáneas es necesario hacer frente al problema de la inestabilidad. La detección de comunidades en general es un proceso inestable, no sólo porque muchos de los mejores algoritmos que se conocen sean estocásticos, sino también porque pequeños cambios dentro de una red pueden llevar a cambios enormes en la estructura de comunidad [2, 48]. Esto provoca que, al detectar cambios entre dos instantáneas consecutivas, no sea trivial discernir si dichos cambios se deben a cambios reales de la red o a la naturaleza inestable del problema. Por otro lado, los métodos que usen redes temporales se tienen que asegurar de mantener una coherencia a largo plazo. Por su diseño e implementación, usar redes temporales implica actualizar una solución de manera iterativa y, por lo tanto, es posible que en redes muy longevas surja una divergencia entre la topología de la red en un instante concreto y la estructura de comunidad.

Para finalizar, cabe destacar que ambos modelos no son excluyentes entre sí. Pasar de una red temporal a una secuencia de instantáneas es un proceso trivial (aunque hay que tener en cuenta que hacerlo a la inversa no es posible, dado que el proceso de transformación hace que se pierda información por la diferencia de precisión entre los modelos). Por lo tanto, uno no debe restringirse a un único modelo. La mejor estrategia es usar el modelo adecuado para cada una de las tareas que se lleven a cabo al analizar una red dinámica. Nada nos impide comenzar un análisis usando una red temporal, calcular en ella puntos en el tiempo donde se detecten cambios bruscos y, a partir de dichos puntos, crear una secuencia de instantáneas para continuar con el análisis.

2.2. Detección de comunidades

En pocas palabras el CDP se puede definir como el proceso de clustering (o agrupamiento) realizado sobre redes. En la introducción de esta tesis se planteó el problema de una manera informal, en consecuencia, el objetivo de esta sección es profundizar en el CDP presentando una definición formal de dicho problema. Se comenzará definiendo el CDP en su versión estática para, a continuación, pasar a definir su versión dinámica.

2.2.1. Detección de comunidades estáticas

Dado un grafo $G = (V, E)$, el problema de detección de comunidades consiste en agrupar los nodos de dicho grafo $V = \{v_1...v_n\}$ en clusters o comunidades $C_i = \{v_0...v_k\}$ teniendo en cuenta las aristas $E = \{e_1...e_n\}$ de este . Al conjunto de todas las comunidades de un grafo $CS = \{C_1...C_m\}$ se le conoce como su estructura de comunidad. Cuando un nodo del grafo puede pertenecer a más de una comunidad a la vez ($\bigcap_{C_i \in CS} C_i \neq \emptyset$) hablaremos de comunidades *solapdas*. Sin embargo, por el contrario, cuando un nodo solo puede pertenecer a una única comunidad ($\bigcap_{C_i \in CS} C_i = \emptyset$) hablaremos de comunidades *no-solapadas*.

Para una revisión exhaustiva de la detección de comunidades en grafos remitimos al lector a [90], donde se describen y comparan diferentes métodos de detección de comunidades diseñados

para trabajar con diferentes tipos de grafos (ponderados, dirigidos, no dirigidos ... etc). Además, se categorizan y estudian las diferentes familias de algoritmos que existen para detectar comunidades, entre las que caben destacar los métodos por corte, por análisis espectral o por análisis del grado de los nodos entre otros, y la complejidad computacional de estos.

2.2.2. Detección de comunidades dinámicas

En el estado del arte se pueden encontrar diferentes tipos de análisis bajo el paraguas de la detección de comunidades dinámicas. Por lo tanto, el primer paso para comprender bien el área es conocer dichos tipos. A grosso modo se pueden identificar tres tipos de análisis, a saber: *rastreo de comunidades*, *historia de las comunidades* y *detección de cambios*.

- El **rastreo de comunidades**, es el análisis más similar al que podemos encontrar en la versión estática del problema y consiste en generar una única división de comunidades para toda la red temporal. En este análisis, dada una red dinámica DN, ya sea modelada como una secuencia de instantáneas o una red temporal, el problema consiste en encontrar una única estructura de comunidad $CS = \{C_1...C_n\}$ para dicha red temporal. Este problema se puede abordar desde dos angulos diferentes. Por un lado, hay métodos que tratan de buscar una única estructura de comunidad CS que se ajuste bien a la red en en todos los momentos de su vida a la vez. Esta visión se asemeja a la seguida en los problemas Robust Optimization Over Time (Optimización robusta duradera en el tiempo) (ROOT) [116], donde se busca una solución duradera en el tiempo en un entorno dinámico. Por otro lado, hay métodos que tratan de obtener una estructura de comunidad que corresponda a la red en el último momento temporal del que se tiene constancia. En esta categoría se encuentran métodos que, por lo general, son capaces de actualizar la estructura de comunidad de manera muy rápida, analizando los cambios uno a uno de manera iterativa. Aunque también se pueden encontrar, en menor medida, métodos que no realizan este proceso de manera iterativa y analizan una red temporal completa de una sola tacada.

- Si el análisis del rastreo de comunidades consistía en generar una única división de la red, el análisis de la **historia de las comunidades** consiste en generar varias estructuras de comunidad para la misma red dinámica. En este análisis, dada una red dinámica DN, ya sea modelada como una secuencia de instántaneas o una red temporal, el problema consiste en obtener un conjunto de comunidades dinámicas $DCS = \{DC_1...DC_n\}$ donde, a su vez, cada elemento de dicho conjunto $DC_i = \{C_i^1...C_i^t\}$ es una secuencia de la misma comunidad C_i en distintos momentos temporales. Por así decirlo, cada DC_i representa un histórico de lo que ha pasado con la comunidade C_i durante la evolución de la red. Este histórico también es conocido como el ciclo de vida de las comunidades, ya que nos indica qué pasa con una comunidad desde que aparece por primera vez hasta que muere. Para definir el ciclo de vida de una comunidad, Palla et al. identificó seis posibles eventos (Crecimiento, Encogimiento, División, Unión, Nacimiento y Muerte) [73]. Estos seis eventos fueron luego extendidos a siete por Cazabet et al. al añadir el evento Resurgimiento [18]. La Figura 2.2 muestra un diagrama de los siete eventos para su mejor comprensión. En el estado del arte, el análisis de la historia de las comunidades se suele dividir en dos etapas, aunque no es estrictamente necesario y pueden surgir métodos que hagan ambos pasos a la vez. En una primera etapa, se calculan las comunidades para los diferentes instantes de la red y en una segunda, se calculan los eventos que ocurren entre los mismos.

- Por último, la **detección de cambios**, se diferencia de los dos análisis anteriores en que no se genera una estructura de comunidad per se, sino que se detectan momentos durante la vida de una red temporal en la que surgen cambios bruscos. Es decir, dada una red dinámica modelada como una red temporal $TN = (V, E^T)$, el objetivo es encontrar una conjunto de etiquetas temporales $TL = \{t_1...t_T\}$ donde cada t_i indica un momento en el tiempo donde ha ocurrido un cambio brusco. De manera implícita estos métodos dan a entender que la red se puede considerar estable entre dichos puntos. Este tipo de análisis se suelen aplicar como un paso previo a los análisis descritos anteriormente. Nótese que aunque no hay ningún impedimento teórico que evite hacer este tipo de análisis con redes dinámicas modeladas como secuencias de instántaneas, no tiene mucho sentido utilizarlas para realizar este tipo de análisis, ya que sería sobrecomplicar el problema de manera innecesaria.

En la literatura se han propuesto varias taxonomías para clasificar los métodos de búsqueda de comunidades dinámicas. Sin embargo, en este trabajo utilizaremos una diferente. Un diagrama conceptual de la taxonomia propuesta se puede encontrar en la Figura 2.3. Tal como muestra la figura, en primer lugar, los métodos se clasificarán según el modelo de red dinámica utilizado. A continuación, estos métodos se categorizarán de nuevo de acuerdo con el tipo análisis que hagan. Veáse que hay algunos tipos de análisis que aparecen repetidos para los dos modelos de red disponibles. Esto significa que se pueden realizar usando cualquiera de los dos modelos disponibles.

- **Secuencia de instantáneas**
  - **Ratreo de comunidades**: Los métodos de esta categoría se centran en encontrar una estructura de comunidad que se ajuste a todas las instantáneas de la red. Estos métodos procesan todas las instantáneas de la red simultáneamente en un único proceso, encontrando comunidades no sólo compuestas por nodos de la misma instantánea sino también por nodos de otras instantáneas diferentes. Algunos trabajos relevantes en esta categoría son [97, 65, 112, 62, 106, 59].
  - **Historia de las comunidades - detección de eventos**: Estos métodos caracterizan una comunidad como una secuencia de eventos que describen que les ocurre entre instantáneas consecutivas. Para poder funcionar necesitan conocer la estructura de comunidad en cada instantánea de la red, la cuál debe ser calculada previamente usando otros métodos. Algunos trabajos relevantes en esta categoría son [73, 3, 39, 12, 94, 104, 95].
  - **Historia de las comunidades - detección de la estructura de comunidad**: Estos métodos se centran en encontrar una estructura de comunidades adecuada para cada instantánea de la red, procesando las instantáneas en su orden natural y utilizando los resultados encontrados previamente para guiar el procedimiento. Mientras que algunos métodos de esta subcategoría tratan de estabilizar la estructura de la comunidad a lo largo del tiempo [19], otros tratan de reducir el cómputo requerido [76, 74]. Algunos trabajos relevantes en esta categoría son [19, 96, 57, 33, 58, 38, 1, 121].
- **Redes temporales**
  - **Detección de cambios**: los métodos de esta se centran en dividir una red en diferentes períodos homogéneos separados por cambios drásticos, mientras buscan una

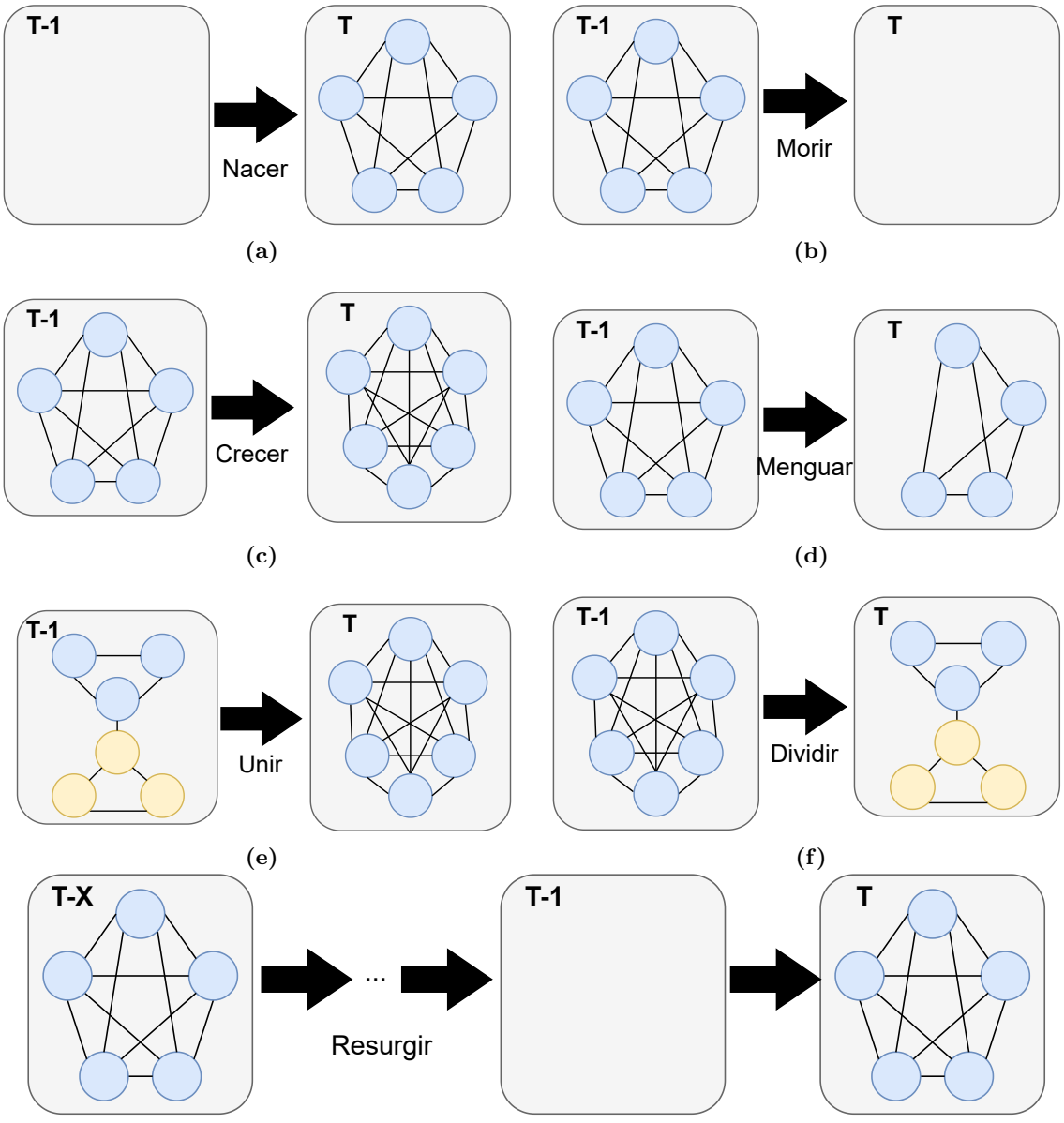


Figura 2.2: *Posibles eventos en el ciclo de vida de una comunidad [73, 18]*

.

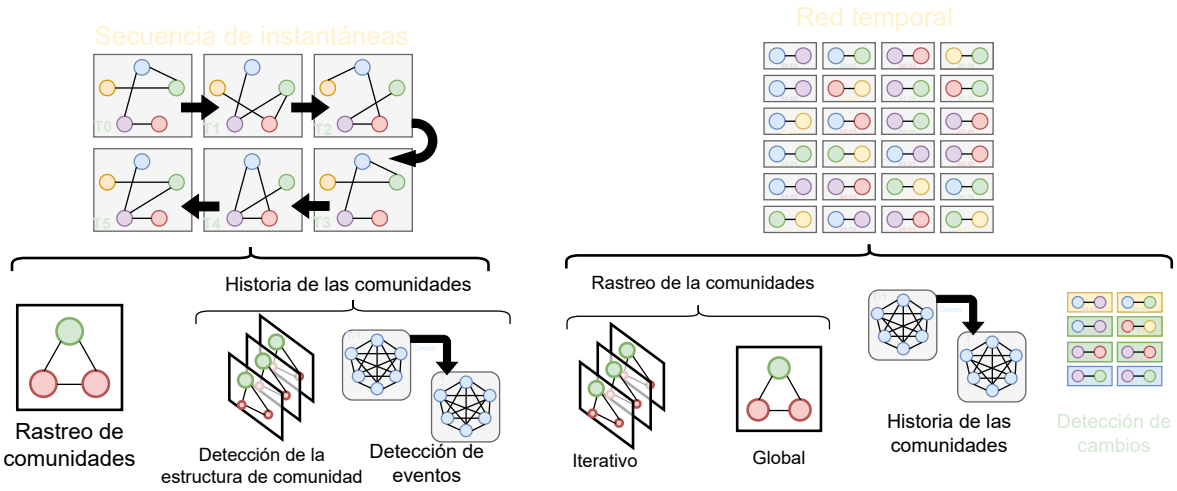


Figura 2.3: *Taxonomia propuesta para clasificar los métodos de búsqueda de comunidades dinámicas.*

estructura de comunidad adecuada para cada periodo. En esencia, estos métodos parten de una red temporal y la transforman en una secuencia de instantáneas. Algunos trabajos relevantes en esta categoría son [93, 20, 30, 4, 79, 46].

- **Rastreo de comunidades global**: los métodos de esta categoría tratan de encontrar estructuras persistentes durante toda la vida de una red dinámica. Algunos métodos requieren fijar una duración para utilizarse como umbral, mientras que otros pueden encontrar estructuras con duraciones arbitrarias. Algunos trabajos relevantes en esta categoría son [101, 43, 56, 102, 6, 83].
- **Rastreo de comunidades iterativos**: los métodos de esta categoría rastrean las estructuras comunitarias de forma iterativa. Cada cambio de la red se procesa de forma continua actualizando la estructura real de la comunidad cuando un cambio lo exige. Esta categoría incluye métodos que definen un conjunto de reglas para actualizar una comunidad dado un cambio y métodos que hacen una optimización dinámica de alguna métrica de calidad. Los trabajos relevantes en esta categoría son [15, 69, 55, 118, 86, 10].
- **Historia de las comunidades**: los métodos de esta categoría adaptan la estructura de la comunidad a los cambios de la red mientras que hacen el seguimiento de los eventos que surgen en las comunidades dinámicas (nacimiento, crecimiento, división ...etc). Algunos trabajos relevantes en esta categoría son [31, 9].

2.3. Medida o evaluación de la calidad de una comunidad

Una parte esencial para el avance de esta área de investigación, está relacionada con las métricas o propiedades que permiten determinar la calidad de una división en comunidades. Sin estas métricas los científicos del área carecemos de un punto común de referencia con el que poder compararnos y así poder medir el progreso o contribución de las nuevas metodologías propuestas. Se ha dedicado mucho esfuerzo en el desarrollo de bancos de pruebas que permitan

comparar entre sí los distintos métodos para la detección de comunidades, aunque la mayoría se han centrado en la variante estática del problema. En consecuencia, en esta sección se comienza haciendo un repaso de las técnicas más comunes para evaluar la calidad de una comunidad para, a continuación, analizar las implicaciones que tiene la variable tiempo a la hora de trabajar con comunidades dinámicas.

2.3.1. Evaluación de comunidades estáticas

Como ya expusimos anteriormente, una estructura de comunidad debe presentar más conexiones entre los miembros de la misma comunidad que entre otros nodos de la red. En el estado del arte se pueden encontrar decenas de métricas capaces de medir esta propiedad, cada una con sus ventajas, incovenientes y limitaciones. Entre ellas podemos destacar:

- *Conductancia*(Co) [47]: fracción del volumen total de aristas que apunta fuera de la comunidad. Definida formalmente como:

$$Co = \sum_{C \in CS} \frac{cs}{2ms + cs}$$

  Dónde $cs = |(u, v) \in E : u \in C \wedge v \notin C|$ es el número de aristas que apuntan fuera de la comunidad y $ms = |(u, v) \in E : u \in C \wedge v \in C|$ el número de aristas dentro de la comunidad.

- *Cut Ratio*(C_{ratio}) [36]: fracción de aristas, de entre todas las posibles, que salen de la comunidad. Definida formalmente como:

$$C_{ratio} = \sum_{C \in CS} \frac{cs}{ns * (n - ns)}$$

  Dónde $cs = |(u, v) \in E : u \in C \wedge v \notin C|$ es el número de aristas que apuntan fuera de la comunidad, $ns = |n : n \in C|$ el número de nodos en la comunidad y $n = |V|$ el número total de nodos en el grafo.

- *Average ODF*(O_{avg}) [32]: fracción media de las aristas de un nodo miembro de una comunidad que apuntan fuera de esta. Definida formalmente como:

$$O_{avg} = \sum_{C \in CS} \frac{1}{ns} * \sum_{u \in C} \frac{|(u, v) \in E : v \notin C|}{d(u)}$$

  Dónde $ns = |n : n \in C|$ el número de nodos en la comunidad y $d(u) = |(u, v) \in E|$ es el grado de un nodo.

- *Community Score*(C_{score}) [80]: mide la densidad de una comunidad basándose en el volumen y las medias de las filas/columnas de la submatriz que la conforma. Definida formalmente como:

$$C_{score} = \sum_{C \in CS} (\frac{2ms}{ns})^2$$

  Dónde $ns = |n : n \in C|$ el número de nodos en la comunidad y $ms = |(u, v) \in E : u \in C \wedge v \in C|$ el número de aristas dentro de la comunidad.

- *Modularidad*(Q) [67]: que mide la diferencia entre la fracción de aristas intracomunitarias de una comunidad respecto al número esperado de dichas aristas si se distribuyen según un modelo nulo (aleatorio). Definida formalmente como:

$$Q = \sum_{C \in CS} (\frac{ms}{m} - (\frac{ms + cs}{2m})^2)$$

  Dónde $ms = |(u, v) \in E : u \in C \wedge v \in C|$ el número de aristas dentro de la comunidad, $m = |E|$ es el número total de aristas en el grafo y $cs = |(u, v) \in E : u \in C \wedge v \notin C|$ es el número de aristas que apuntan fuera de la comunidad.

Sin lugar a dudas, entre las métricas expuestas anteriormente, la *Modularidad* es la métrica más ampliamente utilizada dentro del área para medir la calidad de una estructura de comunidad. No obstante, esta métrica no está libre de limitaciones. En estudios posteriores se comprobó que la *Modularidad* sufre de un problema de resolución ya que tiende a valorar mejor comunidades grandes frente a pequeñas [51]. Por está razón, en la última década se han propuesto alternativas a la *Modularidad* que intentan sobrepasar dicho límite de resolución. Algunas de ellas extienden directamente la *Modularidad* misma, como la *Z-Modularity* [63] o la *Modularity-density* [120], mientras que otras exploran aproximaciones completamente diferentes, como la *Surprise* [99].

Además de las métricas descritas anteriormente, que tratan de medir la calidad de una comunidad a partir de la distribución de sus aristas, otra métrica popular es el Normalized Mutual Information (NMI) [52]. Al contrario que con las métricas anteriores, el NMI no hace uso de la topología de la red, sino que compara cuánto se parece una estructura de comunidad determinada a un *ground truth*. El ground truth es una estructura de comunidad conocida que se sabe a ciencia cierta que es correcta. Las redes con ground truth no son comunes en el mundo real y, aunque se pueden encontrar algunas redes reales con ello, suelen ser escasas. Por esa razón, la comunidad científica se ha volcado en crear algoritmos capaces de generar redes artificiales con las mismas propiedades que las redes encontradas en el mundo real. Estas redes sintéticas son muy versátiles ya que se pueden configurar a conveniencia, incluida la estructura de comunidad, y permiten evaluar cómo funciona y escala un algoritmo. Entre los algoritmos más famosos para generar redes estáticas con estructura de comunidad podemos detacar el LFR [53] y el Stochastic Block Model [45]. Sin embargo, dichas redes sintéticas no dejan de ser modelos y por lo tanto son incapaces de representar en su totalidad las complejidades de una red real.

2.3.2. Evaluación de comunidades dinámicas

Todas las métricas descritas en el párrafo anterior se centran en determinar cómo de bien se ajusta una estructura de comunidad a la topología de una red. Cuando se trabaja en la versión estática del problema, con medir esta propiedad suele ser suficiente. Sin embargo, al pasar a la versión dinámica del problema, medir sólo esta propiedad de la estructura de comunidad suele resultar insuficiente. Por supuesto que es importante asegurarse que la estructura de comunidad concuerda con la topología de una red dinámica en todo momento, pero también es necesario asegurarse de que existe cierta coherencia temporal en la estructura de comunidad. Una estructura de comunidad donde cada comunidad sólo existe en periodos cortos y sin una continuidad no aporta demasiada información. Es por esta razón que Cazabet et al. [16] identifica tres problemas que debemos evitar si queremos tener comunidades dinámicas de calidad, a saber: la *sobre simplificación*, los *glitches* y la *pérdida de identidad*.

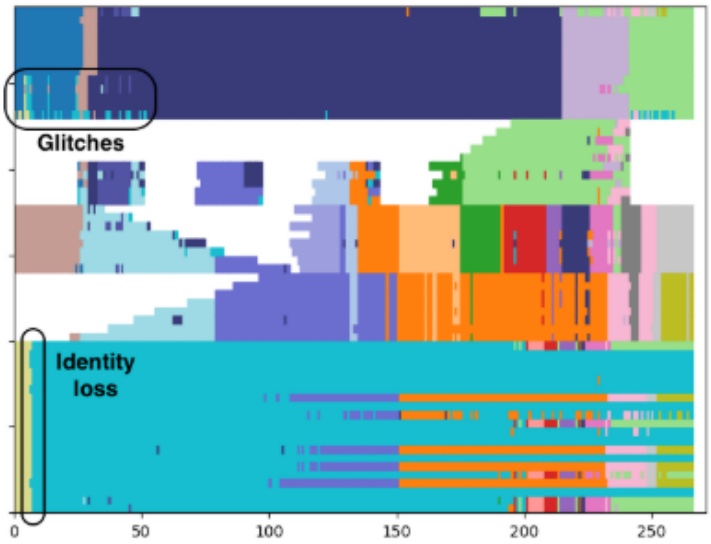


Figura 2.4: *Visualización de una estructura de comunidad dinámica. Cada fila corresponde a un nodo de la red y las columnas representan el tiempo de vida de la misma (cada columna es un instante temporal). El color de los nodos indica la comunidad a la que pertenecen, todos los nodos del mismo color pertenecen a la misma comunidad. Por último se marca un ejemplo de glitch y de pérdida de identidad. Imagen obtenida directamente del articulo [16].*

- La **sobre simplificación**: se da cuando la estructura de comunidad no se ajusta correctamente a la topología de la red en un momento dado. Este fallo conlleva la pérdida de eventos importantes dentro de una red y se puede medir usando cualquiera de las métricas descritas en la parte estática de esta sección (como la *Modularidad* o el NMI).

- Los **glitches**: en el contexto de los videojuegos, un glitch consiste en un fallo gráfico, normalmente la aparición de manchas en la pantalla con colores raros. Cazabet et al. se apoyan en esta analogía para referirse a aquellos nodos que cambian constantemente de comunidad sin una razón aparente, ya que al representar las comunidades de forma gráfica recuerdan a los glitches de los videojuegos (véase la Figura 2.4). Además de identificarse gráficamente se puede medir usando la métrica SM-N [16], cuya definición está disponible a continuación:

$$SM_N = (\sum_{u \in V} \sum_{t=0}^{T-1} \delta(L_{u,t}, L_{u,t+1}))^{-}1$$

  Dónde T es el tiempo de vida total de la red dinámica, $L_{u,t}$ es la etiqueta del nodo u en el instante t y δ es el delta de kronecker ($\delta(x, y) = 1$ si $x = y$ y 0 en caso contrario).

- La **perdida de identidad**: corresponde a la pérdida de la etiqueta de una comunidad y su sustitución por otra diferente mientras que la estructura de comunidad permanece mayoritariamente coherente. Se puede identificar como un cambio de color en una comunidad (véase la Figura 2.4) o medir usando la métrica SM-L [16], cuya definición está disponible a continuación:

$$SM_L = (\sum_{u \in V} H(L(u)))^{-}1$$

  Dónde $H(X)$ es la entropía de Shannon [91] y $L(x)$ es la probabilidad de observar cada etiqueta al elegir el nodo x en un momento aleatorio t.

Por último, al igual que en la variante estática del problema, también se han desarrollado métodos para generar redes sintéticas dinámicas, aunque en menor medida. Entre ellos caben destacar dos: DANCER [54] y el propuesto por Cazabet et al. [16] (a partir de ahora nos referiremos a él como Tnetworks por ser el nombre de la librería en la que está incluido).

2.4. Algoritmos genéticos en el ámbito de la detección de comunidades

En esta sección se estudian los GA en el ámbito de la detección de comunidades. Se comienza describiendo qué es un GA y qué componentes lo conforman. A continuación, se analizan los dos componentes principales que se deben definir para aplicar estos algortimos al CDP, el mecanismo de codificación de los individuos y la función de fitness. Acto seguido, se repasan las peculiaridades que surgen al abordar el CDP desde una prespectiva multi-objetivo y desde una prespectiva dinámica. Por último, se hace un repaso de los principales Multi-Objective Genetic Algorithm (Algoritmos genéticos multi-objetivo) (MOGA) que podemos encontrar en el estado del arte.

2.4.1. Introdución a los algoritmos genéticos

Los Algoritmos Genéticos (GA) son una de las familias más populares y empleadas en el ámbito de las meta-heurísticas bio-inspiradas. Se trata de métodos de optimización poblacionales y estocásticos, que en lugar de mejorar una única solución mejoran un conjunto de ellas, y están basados en los experimentos de Mendel sobre la genética y el concepto darwiniano de la supervivencia de los más aptos. Los GAs ofrecen un compromiso excelente entre potencia, flexibilidad y escalabilidad. Su mejor ventaja es que son capaces de generar soluciones con una calidad aceptable en un tiempo de ejecución acotado, cosa que los métodos clásicos no pueden garantizar. Cuando un problema crece hasta tamaños donde los métodos clásicos dejan de ser aplicables, ya sea por tiempo o memoria, un GA (o en su defecto cualquier meta-heurística bio-inspirada) son candidatos excelentes para obtener soluciones con una calidad aceptable, consumiendo una cantidad de recursos asumible.

Los GAs son una de las familias clásicas entre las meta-heurísticas bio-inspiradas y se llevan estudiando y aplicando por décadas. Fueron propuestos por primera vez por John Holland en la década de los 60 [44], pero no se popularizaron hasta los años 90. La estructura básica de un algoritmo genético se puede ver en la Figura 2.5 y se componen de los siguientes pasos, a saber: *Crear una población inicial*; *Evaluar la aptitud de la población*; *Proceso de selección*; *Cruce y Mutación*; *Remplazo* y por último, *Criterio de parada*. Como ya hemos dicho anteriormente, los GAs son muy flexibles y por lo tanto se pueden aplicar a cualquier área sin tener que codificar una gran cantidad de información sobre el problema. Para poder aplicar un GA a un área concreta necesitamos definir, principalmente, dos componentes: la *codificación* de los individuos y la *función de fitness*.

2.4.2. Representación o codificación de GAs para la detección de comunidades

En el estado del arte podemos encontrar tres codificaciones o representaciones diferentes habitualmente empleadas cuando se utilizar GAs en problemas de CDP, siempre que se busquen

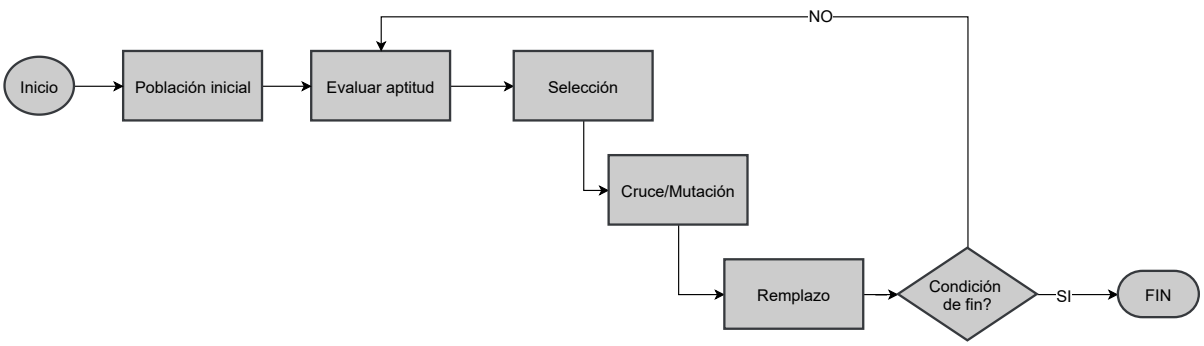


Figura 2.5: *Diagrama de secuencia de un algoritmo génetico genérico.*

comunidades nítidas (aquellas donde los nodos sólo pueden pertenecer a una única comunidad). Estas codificaciones son: la *basada en etiquetas*, la *basada en locus* y la *basada en medoides* [81], que pasaremos a describir a continuación:

- La *codificación basada en etiquetas* está formada por un vector de longitud n de números enteros entre 0 y n, siendo $n = |V|$ el número de nodos de una red. Cada una de las posiciones del vector corresponden a un nodo de la red y todos los nodos que posean la misma etiqueta en el vector pertenecerán a la misma comunidad. La subfigura (b) de la Figura 2.6, muestra un ejemplo de individuo que usa la codificación basada en etiquetas para representar la estructura de comunidad propuesta en la subfigura (a). Tal como muestra la figura, los nodos 1, 2 y 3 pertenecen a la misma comunidad por tener la etiqueta "1". De manera análoga, los nodos 4, 5 y 6 pertenecen a la misma comunidad por tener la etiqueta "2" y los nodos 7, 8, 9 y 10 a otra por tener la etiqueta "3". Las principales ventajas de esta codificación son: la sencillez a la hora de decodificar los individuos y el no necesitar conocer el número de comunidades que queremos buscar de antemano, mientras que, sus principales inconvenientes son: el nivel de redundancia que presenta y la incompatibilidad con los operadores clásicos. Respecto al nivel de redundancia, de manera general, para un grafo con n nodos, suponiendo que no hay ninguna restricción sobre el posible número de comunidades, el espacio de búsqueda disponible es de n^n. Para reducir el tamaño de este espacio de búsqueda se han propuesto procedimientos de renumeración, no obstante, estos conllevan un incremento en el tiempo de cálculo de cada cruce y mutación. Respecto a los problemas de compatibilidad, este tipo de codificación puede generar individuos inválidos en un cruce cuando se usan operadores clásicos. En el área de la detección de comunidades, un individuo se considera inválido cuando coloca en la misma comunidad a dos nodos entre los que no exista un camino que los una. Por esta razón, al usar esta codificación es necesario definir operadores de cruce especializados si trabajamos con problemas donde se pueden dar individuos inválidos.

- La *codificación basada en locus*, al igual que la basada en etiquetas, también se compone de un vector de longitud n de números enteros entre 0 y n. De la misma manera, cada posición del vector también corresponde a un nodo de la red. Sin embargo, los valores de dicho vector, en vez de representar etiquetas, representan a otros nodos de la red, de tal manera que si la posición i del vector contiene el valor j, entonces los nodos i y j pertenecen a la misma comunidad. La subfigura (c) de la Figura 2.6 muestra un ejemplo de individuo que usa la codificación basada en locus para representar la estructura de

comunidad propuesta en la subfigura (a). Los individuos que usan esta codificación son más difíciles de leer, para decodificarlos es necesario hacer un grafo intermedio con n nodos, ir poblándolo con las aristas que aparezcan en el vector y por último calcular las componentes conexas para obtener las comunidades; todos los nodos en la misma componente conexa pertenecen a la misma comunidad. Siguiendo este proceso se generó el grafo que aparece en la subfigura (c). Es justamente este proceso de decodificación intermedio el mayor inconveniente de esta codificación, puesto que, por naturaleza, es menos redundante que la basada en etiquetas ($\prod_{i=0}^{n} k_i$, donde k_i es el grado del nodo i, frente a n^n) y además compatible con los operadores clásicos, siempre que al crear los individuos se les asigne, en la posición correspondiente del vector, uno de los vecinos del nodo que codifica dicha posición.

- Por último, la *codificación basada en medoides* se compone de un vector de longitud k de números enteros entre 0 y n, donde k es el número de comunidades a buscar y n el número de nodos de la red. En esta codificación cada posición del vector representa el medoide de la comunidad, y al igual que en la representación basada en locus, es necesario un proceso de decodificación complejo. De la misma manera que en un proceso de clustering podemos usar la distancia euclídea para decidir a que cluster pertenece cada elemento, es necesario definir una función que mida la distancia entre nodos, por ejemplo la distancia basada en *random walks*, al trabajar con el CDP. La subfigura (d) de la Figura 2.6 muestra un ejemplo de individuo que usa la codificación basada en medoides para representar la estructura de comunidad propuesta en la subfigura (a). La principal ventaja de esta representación es su tamaño reducido ya que también presenta una redundancia alta (cualquier miembro de una comunidad puede ser un medoide); sin embargo, presenta la limitación de tener que conocer el número de comunidades a buscar de antemano.

2.4.3. Funciones de fitness en el ámbito de la detección de comunidades

Cualquiera de las métricas propuestas en la sección 2.3 para medir la calidad de la estructura de comunidad sin utilizar un *ground truth* son candidatas idóneas a *función de fitness*. En el estado del arte podemos encontrar GAs que optimizan alguna de estas funciones, como GA-Net [80] que optimiza el *Community Score*, sin embargo, esto no es lo más común. Como hemos descrito anteriormente, no hay una métrica infalible para medir la calidad de las comunidades ya que estas pueden presentar sesgos. Este hecho, junto con la existencia de métodos voraces eficientes que optimizan algunas de las métricas más populares [23], hacen de los GAs mono-objetivo (aquellos que sólo optimizan una única función de fitness) candidatos poco interesantes para el CDP. Sin lugar a duda, es mucho más común encontrar MOGA (aquellos que optimizan dos o más funciones de fitness a la vez) en el área de la detección de comunidades [81]. Esto no es de extrañar, puesto que si se escoge bien el conjunto de métricas a optimizar, los posibles sesgos de cada métrica se ven mitigados, eludiendo así posibles problemas de resolución [92].

2.4.4. MOGAs en el ámbito de la Detección de Comunidades

En el estado del arte podemos encontrar una gran variedad de MOGAs, cada uno atacando el problema desde una perspectiva distinta. Entre todo los MOGAs propuestos cabe destacar los

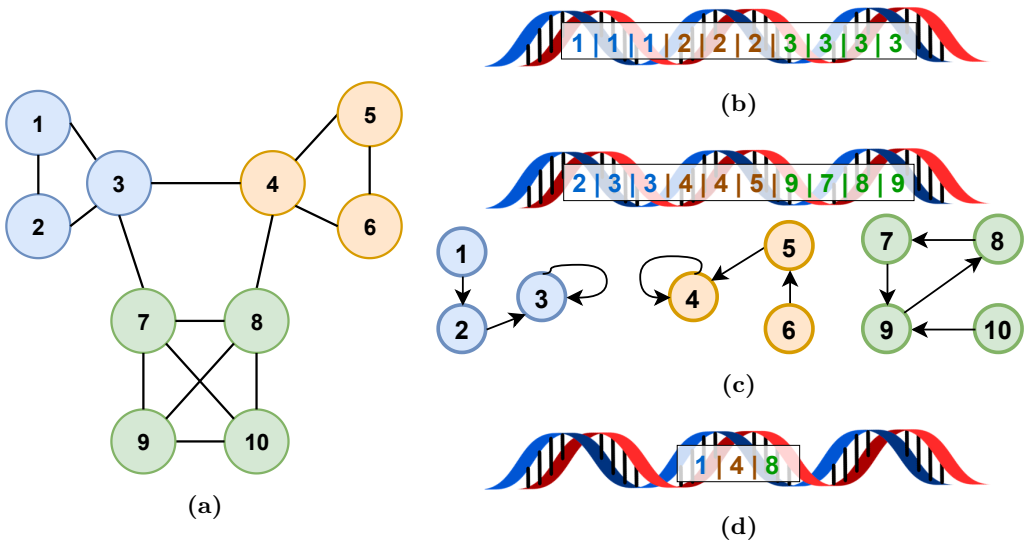


Figura 2.6: *Ejemplos de los diferentes tipos de codificaciones. La subfigura (a) muestra una red con 10 nodos y los colores las comunidades a las que perteneces. La subfigura (b) muestra un ejemplo de individuo que usa la codificación basada en etiquetas para codificar la estructura de comunidad de la subfigura (a); La subfigura (c) muestra un de individuo que usa la codificación basada en locus y debajo ver el resultado del paso intermedio de decodificación. Por último, la subfigura (d) muestra un individuo que usa la codificación basada en medoides.*

siguientes: NSGA-II [28], MOEA/D [119], SPEA2 [123], IBEA [122], HYPE [5] y MoCell [66]. Véase que algunos métodos incluyen un número en su nombre, como NSGA-II o SPEA2. Esto indica que dicho algoritmo es una extensión de una versión anterior. Por ejemplo, aunque en la lista anterior este incluido el algoritmo NSGA-II, ya existe una versión NSGA-III [27], la cual está preparada para trabajar en entornos *many-objective*, es decir, con muchos objetivos (≥ 4), frente a NSGA-II que es recomendable para problemas con 2 ó 3 objetivos como máximo.

Los GAs mono-objetivo, por la propia naturaleza de su diseño e implementación, devuelven una única solución, la mejor encontrada (a excepción de los algoritmos multi-modales, que optimizan una única función de fitness y devuelven más de una solución). Por el contrario, los MOGAs devuelven un conjunto de soluciones. Al trabajar con dos o más funciones objetivo, es posible llegar a soluciones donde una mejora en cualquiera de los objetivos implique un empeoramiento en otro. Este tipo de soluciones, donde la mejora de un objetivo conlleve irremediablemente el empeoramiento de otro, se denominan *no-dominadas* y son justamente las que tratan de buscar los algoritmos multi-objetivo. Este fenómeno hace imposible elegir una única mejor solución y de ahí que los algoritmos multi-objetivo devuelvan un conjunto de soluciones. Al conjunto de todas las soluciones no-dominadas de un problema se le conoce como *Frente de Pareto* (Figura 2.7) y cuanto más se ajuste a dicho frente el conjunto de soluciones no-dominadas encontradas por un algoritmo multi-objetivo, mejor será el rendimiento del mismo. Claro que, el *Frente de Pareto* real de un problema rara vez es conocido y a menudo no queda más remedio que trabajar con aproximaciones. A partir de ahora, por no enredar el lenguaje, no haremos diferencia entre el frente de Pareto real o una aproximación.

[1] `https://commons.wikimedia.org/wiki/File:Pareto_Efficient_Frontier_1024x1024.png`

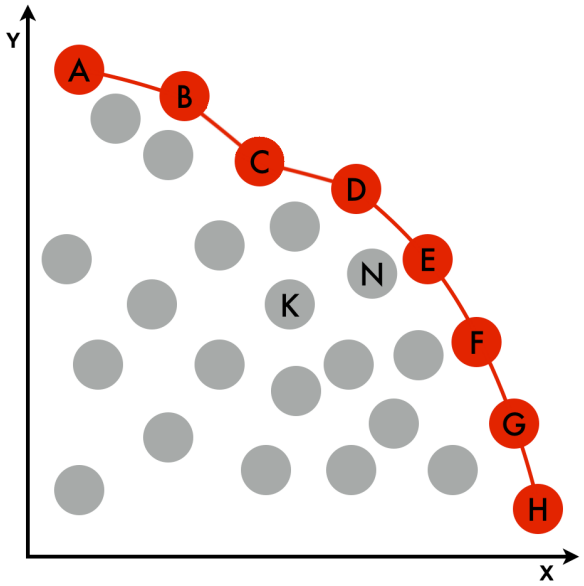


Figura 2.7: *Ejemplo de un frente de Pareto para un problema de maximización bi-objetivo genérico. Cada circulo representa una posible solución y los ejes X e Y el fitness de la solución en cada uno de los objetivos a optimizar. Las soluciones marcadas en rojo (A-H) corresponden al frente de Pareto y las marcadas en gris corresponden al área de soluciones dominadas. Nótese que para cada una de las soluciones marcadas con color gris es posible encontrar una solución en color rojo con mejor fitness en todos los objetivos. Imagen obtenida de wikimedia commons.*[1]

.

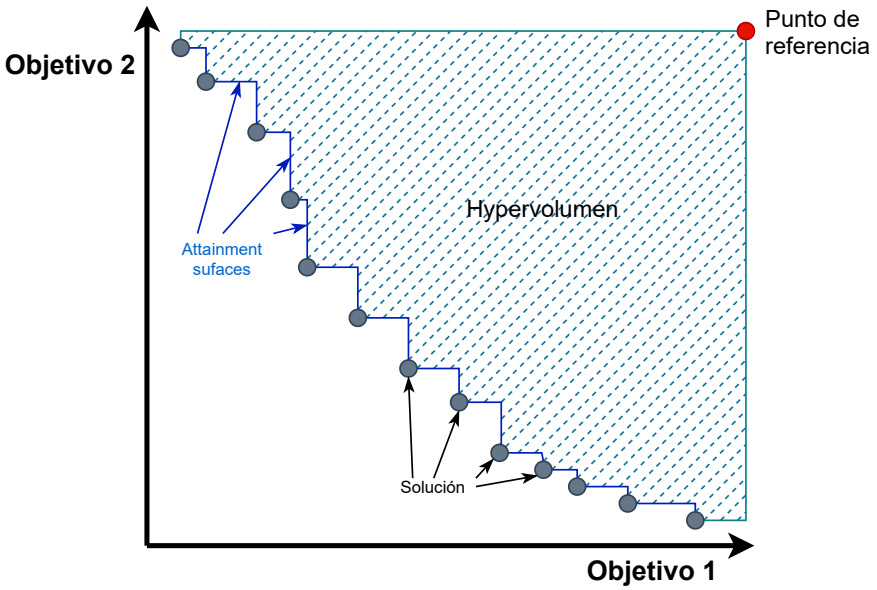


Figura 2.8: *Esquema del cálculo del hypervolumen para un problema de minimización bi-objetivo.*

Se han desarrollado diferentes métricas y metodologías para comparar los Paretos generados por varios algoritmos y así conocer cuál es mejor. Entre ellas caben destacar dos, el Hipervolumen (HV) [124] y las Emperical Attaiment Function (EAF) [60]. El HV mide el hipervolumen, valga la redundancia, que existe entre el conjunto de soluciones no-dominadas y un punto de referencia. La Figura 2.8 muestra un esquema de cálculo del HV para un problema de minimización bi-objetivo. En dicho esquema se puede ver que el HV es equivalente al área que hay entre las *Attiament Surfaces* (líneas azul oscuro), generadas a partir de las soluciones (puntos grises), y el punto de referencia (punto rojo). El HV es muy útil ya que permite resumir en un único valor la calidad del conjunto no-dominado encontrado por un algoritmo multi-objetivo. No obstante, no ofrece información sobre la forma del frente de Pareto, véase el ejemplo de la Figura 2.9. En esa figura se muestra el HV de los frentes obtenidos por dos algoritmos diferentes (el amarillo y el azul). Nótese que el color verde se usa para identificar áreas comunes a ambos algoritmos. Como podemos ver, aunque el área de ambos HV es parecida, las formas son distintas y cada método es capaz de explorar un área del espacio de búsqueda que el otro no alcanza. Las EAF son un complemento perfecto al HV ya que muestran de manera gráfica qué regiones del espacio de búsqueda son capaces de explorar cada algoritmo, siempre y cuando trabaje con 2 o 3 objetivos (el máximo de dimensiones que se pueden plasmar en una representación gráfica).

A la hora de comparar MOGAs, y debido a la naturaleza estocástica de estos, es necesario ejecutar varias veces cada algoritmo para descartar el factor aleatorio o estocástico de los mismos. Este fenómeno hace que la cantidad de datos generados crezca muy rápidamente, lo que dificulta analizar los resultados de forma gráfica. El EAF puede verse como la distribución de la calidad de las soluciones obtenidas por varias ejecuciones de un mismo algoritmo, en pocas palabras, un resumen de los Paretos obtenidos que nos ayude a analizarlos de manera gráfica. Véase el ejemplo de la Figura 2.10. En ella se muestran los resultados obtenidos por 10 ejecuciones de un algoritmo para un problema de minimización bi-objetivo, usando los Paretos en bruto (subfigura a) y las EAFs (subfigura b). En la subfigura (a), cada punto representa las soluciones

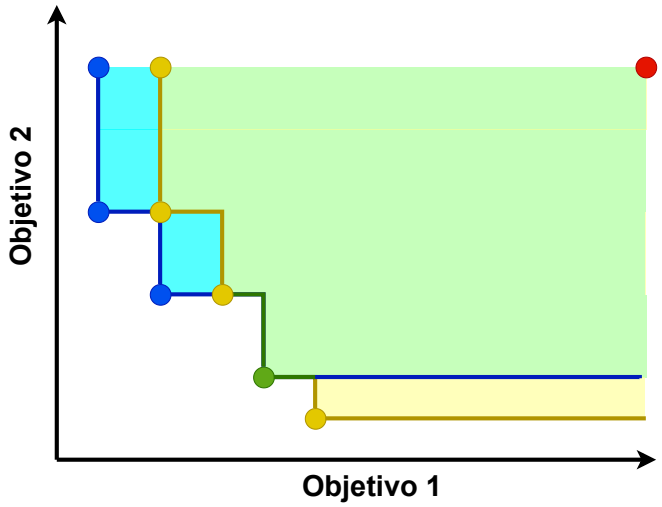


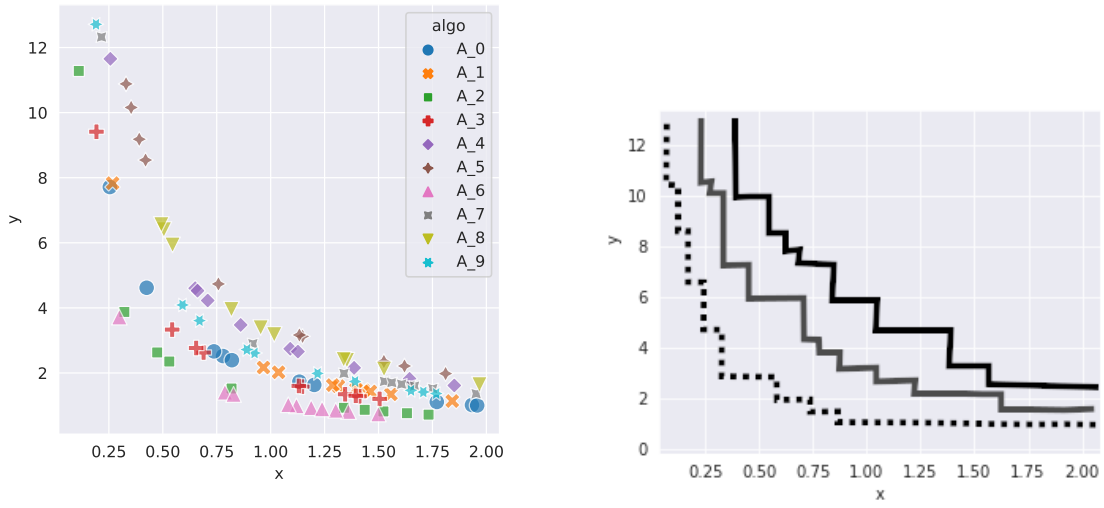
Figura 2.9: *Comparación del Hipervolumen de dos conjuntos de soluciones no-dominadas. En azul las soluciones de un conjunto y en amarillo las del otro. Nótese que el color verde se utiliza para indicar soluciones que ambos conjuntos han encontrado.*

no-dominadas encontradas por el algoritmo y los resultados de cada ejecución se muestran con un color y forma distinta. En la subfigura (b), se muestran las EAFs mejor (punteada), mediana (gris) y peor (negra). En la subfigura (a), discernir qué solución pertenece a cada ejecución es complicado y casi imposible hacerse una idea del rango de soluciones que se ha obtenido y su mediana, mientras que en la (b) esta información está completamente clara. Tengáse en cuenta que en ambas subfiguras, los ejes X e Y representan los objetivos a minimizar.

2.4.5. Algoritmos genéticos en problemas de carácter dinámico

Aplicar un GA en un entorno o problema dinámico no es trivial y presenta dos desafíos extra respecto a los entornos estáticos: *detectar los cambios en el entorno* y *adaptar la población tras dichos cambios* [71]. Detectar un cambio no plantea un problema añadido cuando la llegada de estos se conoce, como en el CDP. Téngase en cuenta que los dos modelos de red dinámica propuestos, la secuencia de instántaneas y la red temporal, contienen información de cuando ocurren las interaciones dentro de la red. Sin embargo, esto si plantea un desafío considerable para aquellos problemas donde el GA tiene que detectar los cambios. La estrategia predilecta para detectar cambios en el entorno es monitorizar el fitness de una población para localizar caídas en el mismo [71], ya que estas caídas son buenos indicadores de que ha habido un cambio en el entorno.

Una vez detectado un cambio, es necesario adaptar el GA a dicho evento, ya que, por norma general, los GAs en su versión estándar no son capaces de adaptarse a un cambio por sí solos. Una forma trivial de tratar dicho problema es descartar la población cuando se detecte un cambio y comenzar de nuevo el proceso evolutivo con una población aleatoria completamente nueva. Este proceso es tremendamente costoso (tanto computacionalmente como en tiempo) ya que todo el trabajo hecho con anterioridad es descartado, por lo que obliga a repetir muchos cálculos que probablemente ya se hayan hecho. Cuando trabajamos en entornos dinámicos, tan importante es encontrar soluciones de calidad dentro del espacio de búsqueda como rastrearlas rápidamente



(a) **(b)**

Figura 2.10: *Gráficas de los Paretos obtenidos por 10 ejecuciones de un mismo algoritmo para un problema de minimización bi-objetivo, donde los ejes X e Y representan los objetivos a minimizar. La subfigura (a) muestra los 10 paretos a la vez y la (b) las EAFs mejor, mediana y peor de los mismos resultados. En la subfigura (a), cada punto representa una de las soluciones no-dominadas encontradas por el algoritmo. Los resultados de cada ejecución se muestran con un color y una forma distinta. En la subfighura (b), la línea negra representa la peor EAF y la gris y la punteada la mediana y la mejor.*

tras un cambio.

A la hora de adaptar un GA a cambios del entorno hay que hacer un análisis a dos niveles distintos, *poblacional* y *individual*.

2.4.5.1. Análisis a nivel poblacional

Téngase en cuenta que, por diseño, un GA comienzan con una población muy diversa (comúnmente con soluciones creadas al azar) que se va especializando hacia las soluciones más prometedoras según avanza el proceso de evolución. Este proceso de convergencia plantea un problema cuando trabajamos en un entorno dinámico. Por un lado, necesitamos que las poblaciones converjan hacia soluciones prometedoras, pero por otro, también queremos que las poblaciones mantengan su diversidad cuando aparezcan cambios en el entorno y así no perder capacidad de búsqueda. En el estado del arte podemos encontrar dos aproximaciones a la hora de adaptar una población tras un cambio [71]. En la primera aproximación, la población se ve afectada sólo cuando ocurre un cambio, mientras que en la segunda, la población sufre cambios constantemente durante todo el proceso de evolución. Independientemente de la aproximación escogida, en el estado del arte también podemos encontrar cuatro familias de técnicas para adaptar una población tras un cambio, a saber, la *hipermutación*, los *esquemas de memoria*, los *esquemas de inmigrantes* y los *esquemas multi-población*. Estas técnicas pretenden aprovechar el trabajo previo lo mejor posible, adaptando una población a un cambio de tal manera que se mantenga un equilibrio entre sus capacidades de exploración/explotación. Nótese que estas cuatro familias de técnicas son combinables y no hay que interpretarlas como familias disjuntas. Es más, sin ir más lejos, Elite Immigrant Genetic Algoritm (Algoritmo genético con inmigrantes elitistas) (EIGA), una de las técnica de inmigrantes más populares, es una combinación entre una técnica de inmigrantes y una técnica de memoria.

- **Hipermutación**: es una de las técnicas más sencillas y consiste en alterar, cuando se detecta un cambio, una o varias de las probabilidades que gobiernan el ciclo evolutivo de un GA, como la probabilidad de mutación [70], probabilidad de cruce [84], elitismo [84] ... etc.
- **Esquemas de memoria**: consisten en almacenar individuos completos o partes de ellos en una memoria para ser reutilizados posteriormente. Podemos encontrar dos aproximaciones a esta técnica: la *implícita* [68, 108], qué incluye la memoria en la codificación de los individuos, y la *explícita* [8, 64, 115], donde la memoria es externa al proceso evolutivo.
- **Esquemas de inmigrantes**: consite en introducir individuos ajenos a la población del GA para aumentar la diversidad/explotación. Hay tres familias de estrategias de inmigrantes: la RIGA [98], donde los individuos insertados son completamente aleatorios; la EIGA [109], donde los individuos insertados son individuos de pasadas generaciones; y la Hybrid Immigrant Genetic Algoritm (Algoritmo genético con inmigrantes híbridos) (HIGA) [111], donde se insertan ambos tipos de individuos (aleatorios o de generaciones pasadas).
- **Esquema multi-poblacíon**: consisten en mantener varias poblaciones en paralelo durante el proceso evolutivo. Estas poblaciones pueden buscar en áreas distintas del espacio de búsqueda [37, 103] u ocuparse de tareas completamente diferentes [11, 100](una población se centra en buscar soluciones prometedoras para el estado actual del problema y otra se anticipa a posibles cambios explorando el espacio de búsqueda)

2.4.5.2. Análisis a nivel individual

En entornos dinámicos es necesario analizar cada individuo de la población de manera independiente, pues en determinados problemas la aparición de un cambio en el entorno puede generar que los individuos de la población dejen de ser válidos, ya sea porque se encuentre en una zona del espacio de búsqueda prohibida o porque rompan alguna restricción del problema. En esos casos es necesario definir mecanismos que transformen estos individuos inválidos en individuos válidos de nuevo, es decir, que los reparen. Al igual que cuando adaptamos la diversidad de la población, es importante que las estrategias de reparación aprovechen el trabajo pasado y que, en la medida de lo posible, el fitness de la población se vea alterado lo mínimo posible. Por desgracia, este fenómeno es muy dependiente del problema sobre el que se aplique el GA e incluso de la codificación que este utilice y por lo tanto no hay recetas genéricas en el estado del arte. Para las estrageias de reparación es necesario buscarlas o, en el peor de los casos, crearlas para nuestro problema ad-hoc.

2.4.6. Algoritmos genéticos para la detección de comunidades dinámicas

Las metaheurísticas bio-inspiradas se han usado ampliamente para la detección de comunidades dinámicas, en especial los MOGAs. En concreto se han usado siguiendo la metodología del *evolutionary clustering* [19] planteada como un problema de optimización bi-objetivo, donde uno de los objetivos se centra en ajustar las comunidades a la topología actual de la red, mientras que el otro fuerza a que las comunidades no se desvíen demasiado de su historia reciente. F. Folino y C. Pizzuti [34], C. Guoqiang et.al [21], B.A. Attea y K. Haider [7] y M. Jingjing et.al [61] han propuesto MOGAs que siguen la metodología anterior. Los dos primeros se basan en NSGA-II y los dos últimos en MOEA/D. Fuera de los algoritmos genéticos también se han propuestos otras metaheurísticas bio-inspiradas, por ejemplo, Y. Ying et.al [114] propusieron un Particle Swarm Optimization (Optimización basada en nubes de partículas) (PSO) [82] con operadores de inicialización y cruce personalizados y Z. Xu et.al [121] propusieron un algoritmo de murciélagos [113]. Sin embargo, todos los métodos anteriores definen criterios triviales para seleccionar una única solución del frente de Pareto; en concreto, se quedan con aquella que mejor se ajuste a la topología actual de la red, lo que favorece a un objetivo frente a otro. La mejor forma de tomar dicha decisión sigue siendo, a día de escribir este documento, una pregunta abierta en el área que merece la pena ser investigada.

A pesar de que se han planteado varios MOGAs para detectar comunidades dinámicas, la adaptación de estos al entorno dinámico para hacerlos eficientes apenas se ha explorado. Todos los métodos mencionados en el párrafo anterior utilizan la estrategia trivial de descartar la población y comenzar el proceso evolutivo de cero cada vez que llega un cambio a la red. Aunque esto no inválide la calidad de las soluciones que encuentran, más bien al contrario, los hace poco aplicables a problemas reales debido a los recursos (en tiempo/memoria) que requieren. Es por esa razón, que esta tesis se centra en dar un primer paso en esa dirección y se plantea estudiar cuáles de las técnicas del estado del arte son las más adecuadas para adaptar las poblaciones de un GA en el CDP dinámico, además de proponer técnicas concretas para reparar individuos. El foco principal de esta tesis es adecuar los GA para trabajar en CDP dinámicos del mundo real.

Capítulo 3

Técnicas eficientes para adaptar individuos a cambios en la red

En este capítulo se proponen tres operadores de reparación para usar en algoritmos genéticos que resuelvan el CDP dinámico usando la codificación basada en locus. Primero se introduce la necesidad de estos operadores de reparación y el trabajo previo necesario para comprenderlos (sección 3.1). A continuación, se describen en detalle los operadores en sí (sección 3.2), para después presentar una serie de experimentos que se han llevado a cabo para probar la validez de dichos operadores (sección 3.3). Finalmente, se exponen las conclusiones obtenidas de dichos experimentos, así como las limitaciones de estos y futuras líneas de investigación (sección 3.4). Todos los resultados de este capítulo fueron publicados en la revista Future Generation Computer Systems (FGCS) en septiembre de 2020 [77]. Una versión de dicho articulo está disponible, sin el formato de la revista, en el anexo R2 de este tomo.

3.1. Introducción

Para poder aplicar un GA a un problema concreto, es necesario definir una codificación de tal manera que cada individuo de la población represente una posible solución a dicho problema. Lo ideal es definir codificaciones dónde sea imposible crear individuos que representen soluciones imposibles o sin sentido. Sin embargo, no es común encontrar codificaciones que cumplan esta propiedad en el mundo real y, a menudo, es necesario adaptar los operadores de inicialización, cruce y/o mutación para que se evite generar soluciones imposibles. En problemas estáticos basta con tener en cuenta los tres operadores anteriores; pero, al pasar a un entorno dinámico, también hay que tener en cuenta los cambios que puedan surgir durante el proceso evolutivo. La llegada de un cambio puede producir que individuos que codificaban soluciones completamente válidas se hayan convertidos en individuos que codifiquen soluciones sin sentido. Por esta razón, si esta casuística se da en un problema dinámico, es necesario definir un operador que sea capaz de transformar los individuos invalidados, tras un cambio, en soluciones válidas de nuevo. Dicho operador es conocido en el área como operador de *reparación*.

Para cada problema concreto pueden existir muchos posibles operadores de reparación; sin embargo, no todos son igual de buenos. A la hora de diseñar un operador de reparación es

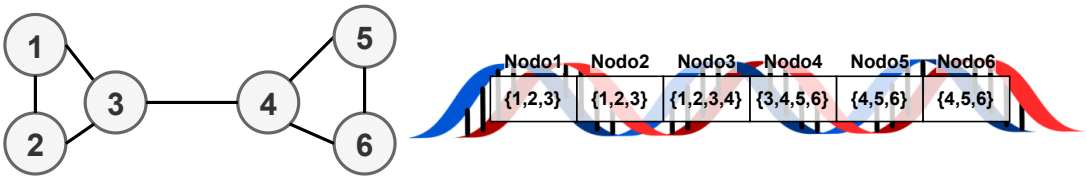


Figura 3.1: *Ejemplo de las restricciones que impone la codificación basada en locus. Esta figura muestra en la derecha una red y en la izquierda los valores que pueden tener cada una de las posiciones del array de genes que conforma un individuo. Cualquier individuo que tenga un valor fuera de los permitidos en cualquier posición del array será considerado un individuo inválido.*

deseable que su aplicación perturbe lo mínimo posible la calidad de las soluciones. Un buen operador de reparación debe ser capaz de aprovechar la información del entorno y así tomar una decisión, lo mejor informada posible, sobre cómo reparar una solución. De poco sirve un operador de reparación si tras su aplicación las soluciones tienen una calidad semejante a la calidad esperada de una solución aleatoria.

Centrándonos en el problema estudiado en esta tesis, el CDP, una solución se puede considerar inválida, de manera general, si coloca en la misma comunidad dos nodos sin un camino que los una. No obstante, dependiendo de la codificación usada este criterio puede ser más restrictivo. Por ejemplo, en la codificación basada en locus, descrita en el Capítulo 2, se estipula que cada posición del vector de genes que conforman un individuo solo puede tener como valor el nodo en cuestión o un vecino de este (Véase la Figura 3.1 para un ejemplo aclaratorio). Si seguimos esta regla a la hora de diseñar un GA, podemos asegurar que jamás se generará un individuo inválido siempre y cuando trabajemos en la versión estática del problema, puesto que, en la versión dinámica no podemos asegurar que un cambio de la red no tenga como consecuencia la ruptura de dicha regla. Véase la Figura 3.2, dónde se muestra un ejemplo de cómo un cambio en una red invalida un individuo. Fíjese qué, en el instante T-1, el individuo de la figura codifica una solución válida, ya que todas las posiciones tienen como valor el nodo en si o un vecino de este. Sin embargo, en el instante T ese mismo individuo codifica una solución inválida, ya que en la cuarta posición contiene un nodo que no es vecino del Nodo 4.

3.2. Diseño de los operadores

Siguiendo en el contexto del CDP, un operador de reparación entonces se puede definir cómo un método que, dado un individuo, primero actualice su tamaño (añada o elimine nodos según haya variado la red) y, a continuación, compruebe si cada uno de sus genes contiene un valor válido, es decir, contenga el nodo en cuestión o un vecino de este, sustituyéndolos en caso de no ser así. El Algoritmo 1 muestra un pseudocódigo que implementa está función. Nótese que en la línea 8 del algoritmo se define un método genérico llamado *seleccionarVecino*. Este método es la clave del operador de reparación, ya que de su eficiencia depende la calidad de un individuo reparado. En su forma más simple, dicho operador se puede definir como un método dónde cada gen inválido se sustituya por un vecino escogido de manera aleatoria. Sin embargo, esta solución tan simple tiene una aplicabilidad limitada, tal como se comprobó en [75]. Este operador no aprovecha el trabajo realizado hasta el momento, por lo que con cada valor reparado se pierde información. En la práctica, aplicar este operador en redes con una tasa de variación ligeramente

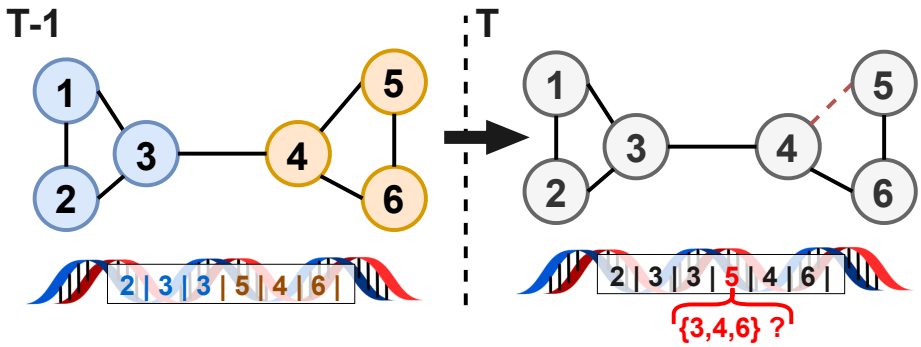


Figura 3.2: *Ejemplo de cómo un cambio en la red invalida un individuo. La imagen muestra una red temporal en dos instantes consecutivos T-1 y T. La arista punteada de color rojo en la red indica que dicha arista desapareció. Además de la red, se muestra un individuo que codifica una solución válida en el momento T-1 y como el mismo individuo ya no es válido en el momento T.*

alta es equivalente a generar una población aleatoria completamente nueva. No es menos cierto que cualquier operador de reparación sufrirá un efecto parecido, cuanto más cambie la red entre dos instantes de tiempo más difícil será escoger valores correctos durante la reparación; pero tampoco es menos cierto que en la reparación al azar este efecto sea demasiado agudo para poderse aplicar a un problema real. En consecuencia, esta sección propone tres operadores de reparación, que aprovechándose de la estructura de comunidad de un instante anterior, son capaces de tomar una decisión informada sobre qué valor escoger a la hora de reparar un gen y así mitigar el problema que presenta la reparación aleatoria.

Algoritmo 1: Reparar población

```
Data: Poblacion, Red, Comunidades
Result: Poblacion
P ← ∅;
forall i ∈ [0 .. longitud(Poblacion)] do
    Ind ← Poblacion[i];
    actualizaLongitud(Red, Ind);
    forall g_i ∈ [0 .. longitud(Ind)] do
        g_v ← Ind[g_i];
        if !g_v ∨ !estanConectados(g_i, g_v, Red) then
            Ind[g_i] ← seleccionarVecino(g_i, Red, Comunidades);
    P ← P ∪ Ind;
Poblacion ← P;
```

3.2.1. Reparación mediante label propagation

El primer operador propuesto está basado en el algoritmo clásico de *Label Propagation* [25]. En dicho algoritmo, cada nodo empieza recibiendo una etiqueta única que se irá propagando por la red de manera iterativa. En cada iteración, se van seleccionando nodos al azar y cada uno

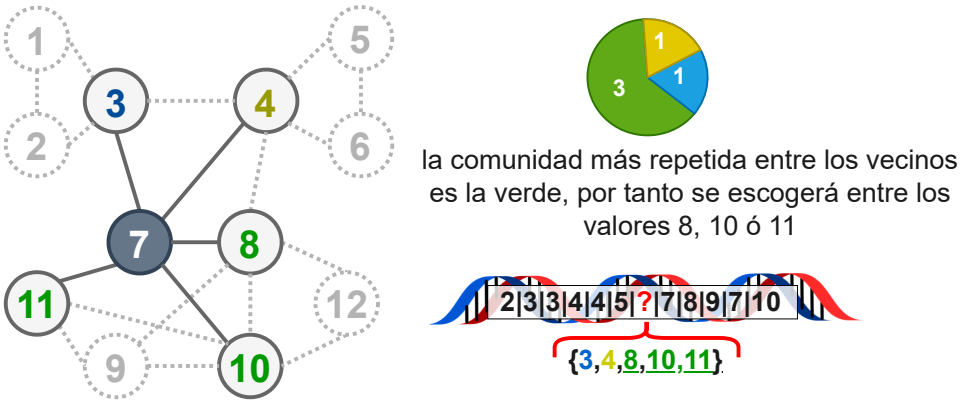


Figura 3.3: *Ejemplo de reparación usando el método de label propagation. Las partes de la red con una línea continua son aquellas que participan en la reparación del nodo, mientras que las punteadas son ignoradas por el operador. El color del número de cada nodo indica la comunidad a la que pertenecía dicho nodo antes de que la red sufriera un cambio.*

heredará la etiqueta más común entre sus vecinos; en caso de empate se elegirá una etiqueta al azar entre las más comunes. El algoritmo termina cuando no se cambia ninguna etiqueta durante una iteración o al alcanzar un número predeterminado de iteraciones. Siguiendo el mecanismo de propagación propuesto por dicho algoritmo, el primer operador de reparación elegirá un vecino al azar que pertenezca a la comunidad más repetida entre los vecinos del nodo a reparar.

El Algoritmo 2 contiene un pseudocódigo que implementa este operador de reparación. En dicho pseudocódigo, primero se identifica la comunidad más repetida entre los vecinos (líneas 1-2), luego se seleccionan los vecinos que pertenecen a dicha comunidad (línea 3) y se elige uno al azar (línea 4). Téngase en cuenta que la función que implementa dicho pseudocódigo sustituiría la función *seleccionarVecino* de la línea 8 del Algoritmo 1. Para finalizar, se ha incluido en la Figura 3.3 un ejemplo del funcionamiento del operador. Esta figura muestra el proceso de reparación del gen correspondiente al nodo 7 de una red. La figura muestra con una línea sólida los nodos involucrados en la reparación y con una punteada el resto de los nodos de la red. El color del número de cada nodo representa la comunidad a la que pertenecía dicho nodo antes de la reparación. Podemos ver que, entre los nodos involucrados en la reparación, la comunidad más repetida es la verde y, por lo tanto, el nodo escogido será cualquiera de los marcados con ese color (8, 10 ó 11).

Algoritmo 2: Reparación mediante label propagation

Data: $Nodo$, Red, $Comunidades$
Result: v

1 $vecinos \leftarrow obtenerVecinos(Node, Red)$;
2 $c \leftarrow masRepetida(vecinos, Comunidades)$;
3 $validos \leftarrow vecinosComunidad(vecinos, c)$;
4 $v \leftarrow elegirAzar(validos)$

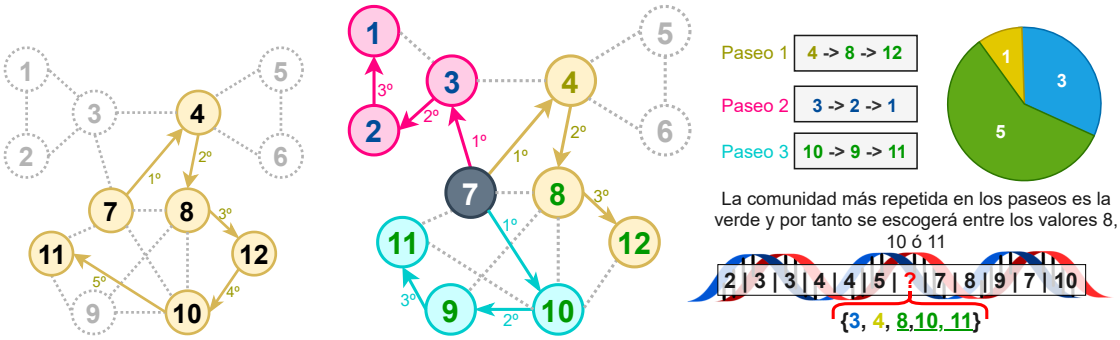
3.2.2. Reparación mediante random walks

El segundo operador propuesto está basado en el algoritmo de *random walks* [88]. Un *random walk* o paseo aleatorio es una técnica muy extendida dentro del área de la detección de comunidades y consiste en elegir un nodo al azar y empezar a recorrer un grafo viajando de nodo a nodo eligiendo aristas también al azar. En la subfigura (a) de la Figura 3.4 podemos ver un ejemplo de un paseo aleatorio de cinco pasos dentro de un grafo. Los nodos y aristas coloreados de amarillo representan el recorrido escogido, mientras que los punteados de blanco representan al resto de la red. El paseo comenzó en el nodo 7 y terminó en el 11 pasando por el 4, 8, 12 y 10 en el orden establecido. La idea detrás de los paseos aleatorios para la detección de comunidades radica en el hecho de que, por estadística, los paseos aleatorios tienden a visitar más nodos de su misma comunidad que a miembros de otras comunidades. Basándose en dicha propiedad, el segundo operador de reparación consiste en ejecutar varios paseos aleatorios, con un número de pasos predeterminado, partiendo del nodo a reparar. Una vez ejecutados los paseos, el operador de reparación comprueba cual es la comunidad que más ha sido visitada durante estos y elige un vecino al azar que pertenezca a dicha comunidad.

El Algoritmo 3 contiene un pseudocódigo que implementa este operador, donde primero se lanzan los distintos paseos aleatorios (líneas 1-4); una vez terminados, se determina la comunidad más visitada durante los mismos (línea 5); y, para finalizar se selecciona un vecino al azar entre los que pertenecían a dicha comunidad (líneas 6-8). Téngase en cuenta, que al igual que con el método basado en *label propagation*, la función que implementa el anterior pseudocódigo viene a sustituir a la función *seleccionarVecino* de la línea 7 del Algoritmo 1. Por último, en la subfigura (b) de la Figura 3.4 se incluye un ejemplo de cómo funciona este operador. La figura muestra un ejemplo de cómo se repara el gen correspondiente al nodo 7 de la misma red mostrada en el ejemplo del reparador basado en *Label Propagation*. En dicha figura podemos ver, marcados con colores distintos, cómo se han ejecutado tres paseos aleatorios, compuestos por tres pasos, comenzando desde el nodo 7. Fíjese que la comunidad más visitada durante dichos paseos es la verde y por lo tanto el método escogerá uno de los vecinos del nodo 7 que pertenezca a dicha comunidad (8, 10 ó 11).

Algoritmo 3: Reparación mediante random walks

```
   Data: Nodo, Red, Comunidades
   Result: v
1  paseos ← ∅;
2  forall i ∈ [0 .. num_caminantes] do
3  |   p_i ← comezarPaseo(Nodo, Red, num_pasos);
4  |   paseos ← paseos ∪ p_i;
5  c ← comunidadMasVisitada(paseos, Comunidades);
6  vecinos ← obtenerVecinos(Nodo, Red);
7  validos ← vecinosComunidad(vecinos, c);
8  v ← elegirAzar(validos)
```



(a) (b)

Figura 3.4: *Ejemplo de un paseo aleatorio y del método de reparación basado en estos. Las partes de la red coloreadas son aquellas que han sido visitadas por un paseo aleatorio, mientras que las punteadas no. El color del número de cada nodo indica la comunidad a la que pertenecía dicho nodo antes de que la red sufriera un cambio.*

3.2.3. Reparación híbrida

El tercer y último operador es una combinación de los dos operadores anteriores denominado *operador híbrido*. La idea detrás de este operador es combinar la potencia de ambos métodos y así poder sobrepasar las posibles limitaciones de ambos. El Algoritmo 4 muestra un pseudocódigo que implementa dicho operador. Nótese que, a diferencia de los pseudocódigos propuestos para los otros dos operadores, este no viene a sustituir la función *seleccionarVecino* de la línea 7 del Algoritmo 1, sino que sustituye dicho algoritmo en su totalidad. El algoritmo propuesto mantiene dos poblaciones diferentes (líneas 1-2), una reparada con el operador basado en *label propagation* y otra reparada con el de *random walks*, que va poblando con los individuos reparados con el operador correspondiente (líneas 3-13). Por último, se calcula el conjunto de soluciones no-dominadas resultado de aplicar la clasificación no-dominada propuesta en el algoritmo NSGA-II [28] a la unión de las dos poblaciones anteriores (línea 14).

3.3. Caso de uso: CDP dinámicas en redes sintéticas y en Travian

Para probar la validez de los operadores de reparación estos se han probado en un caso de uso concreto. Dicho caso de uso consiste en implementar un MOGA específico que integre los operadores de reparación y sea capaz de detectar la estructura de comunidad en redes dinámicas modeladas como secuencias de instantáneas, además de aplicarlo a una batería de pruebas compuesta por 66 redes dinámicas sintéticas y 2 reales (generadas a partir de datos extraídos del juego online Travian). Con esta bateria de pruebas se pretende realizar una experimentación triple: (1) evaluar cuál de las tres técnicas de reparación propuestas funciona mejor; (2) comprobar cómo afecta el mecanismo de reparación al funcionamiento interno del MOGA; (3) por último, comparar la eficiencia del MOGA con otros métodos similares del estado del arte.

A continuación, se definen formalmente las particularidades del CDP dinámico que resuelve este caso de uso y se pasa a describir en detalle el MOGA implementado y la batería de pruebas usada en la experimentación.

Algoritmo 4: Reparación mediante la técnica híbrida

```
   Data: Poblacion, Red, Comunidades
   Result: Poblacion
 1 P_lp ← ∅;
 2 P_rw ← ∅;
 3 forall i ∈ [0 .. longitud(Poblacion)] do
 4     I_lp ← Poblacion[i];
 5     I_rw ← Poblacion[i];
 6     actualizaLongitud(Red, I_lp, I_rw);
 7     forall g_i ∈ [0 .. longitud(I_lp)] do
 8         g_v ← I_lp[g_i];
 9         if !g_v ∨ !estanConectados(g_i, g_v, Red) then
10             I_lp[g_i] ← labelPropgation(g_i, Red, Comunidades);
11             I_rw[g_i] ← randomWalks(g_i, Red, Comunidades);
12     P_lp ← P_lp ∪ I_lp;
13     P_rw ← P_rw ∪ I_rw;
14 Poblacion ← unirPoblaciones(P_lp, P_rw);
```

3.3.1. Definición del problema

Sea $SN = \{G^0..G^n\}$ una red dinámica modelada como una secuncia de n instantáneas ordenadas cronológicamente, donde cada instantánea $G^t = (V^t, E^t)$ es un grafo que representa a la red en el instante t y está compuesto por un conjunto V^t de vértices, o nodos, y un conjunto E^t de aristas, o conexiones ente los elementos de V^t. El objetivo es agrupar los vértices de cada G^t de forma que los miembros de un mismo grupo tengan más aristas entre ellos que con el resto de vértices de G^t. En este trabajo nos centraremos en comunidades no solapadas, por lo que cada vértice o nodo de G^t sólo puede pertenecer a un único grupo.

3.3.2. Diseño del algoritmo genético multi-objetivo

La Figura 3.5 muestra un diagrama de flujo del algoritmo genético propuesto. El método recibe una red dinámica modelada como una secuencia de instantáneas y devuelve la estructura de comunidad para cada instante temporal. Tal cómo muestra la figura, el algoritmo está compuesto de tres partes: una primera dónde se crea una población; una segunda dónde se le aplica un proceso evolutivo siguiendo el algoritmo NSGA-II [28]; y, por último, una tercera dónde se elige una solución del frente de Pareto para ser usada como estructura de comunidad de la instantánea. El proceso es el mismo para todas las instantáneas salvo la primera, donde en vez de reparar la población anterior se genera una población de cero completamente aleatoria. A continuación se describirán cada uno de estos pasos en detalle.

1. Para generar una población completamente aleatoria, se crea cada individuo seleccionando para cada gen, el valor correspondiente a un vecino aleatorio del nodo que representa dicho gen. Sería equivalente a aplicar el Algoritmo 1 en una población de individuos vacíos dónde el método *seleccionarVecino* de la línea 8 escogiese cualquiera de los vecinos al azar. Por

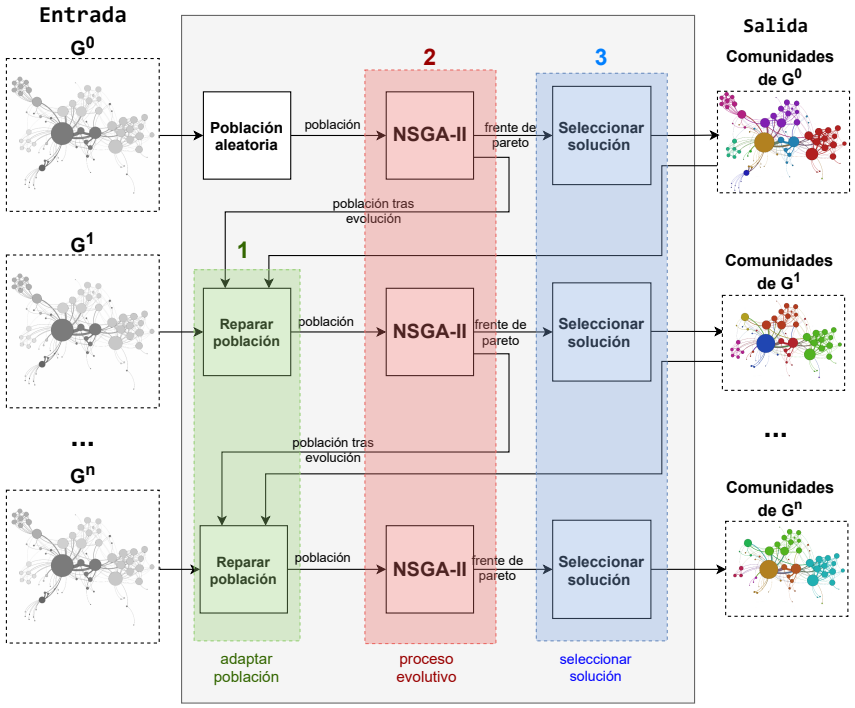


Figura 3.5: *Diagrama del flujo de ejecución del algoritmo evolutivo usado en el caso de uso.*

otro lado, para adaptar la población tras un cambio en la red se usarán las estrategias de reparación propuestas en la sección anterior.

2. Tal como ya se ha dicho, el proceso evolutivo se lleva a cabo usando el algoritmo NSGA-II con individuos codificados con la codificación basada en locus. El Algoritmo 5 contiene un pseudocódigo que implementa el proceso evolutivo. Para el proceso evolutivo, se optimizarán dos funciones de fitness: la inversa del *Community Score* [80] y el *Average ODF* [32] (línea 4), las cuales fueron seleccionadas siguiendo las directrices propuestas en [92] (para más información sobre las funciones remitimos al lector al capítulo 2 de este tomo). Respecto a los operadores evolutivos, para el apareamiento se utiliza el cruce en dos puntos (línea 9); para la selección se realiza un torneo de tamaño 2 (línea 6) y para la mutación se utiliza una función personalizada dónde se cambian un porcentaje de los genes de un individuo por vecinos del nodo correspondiente seleccionados al azar (Algoritmo 6). Finalmente, como criterio de parada del proceso de evolución (línea 3), se utiliza el HV [124] del frente de Pareto. El algoritmo terminará si se alcanza un número máximo de generaciones o si el hipervolumen permanece inalterado durante un número N de generaciones (este valor se fijará experimentalmente).

3. Por último, para seleccionar una única solución del frente de Pareto como la estructura de comunidad de una instantánea se escogerá la solución del frente con la mayor Modularidad [67].

Algoritmo 5: Proceso evolutivo

Data: $Poblacion$, Red, N, MAX_GEN, P_CRX, P_MUT
Result: $Poblacion$

```
1  gen ← 0;
2  hv ← ∅;
3  while gen < MAX_GEN ∧ variaHV(hv, N) do
4      Poblacion ← evaluar(Poblaciopn, Red);
5      hv ← update(hv, Poblacion);
6      Padres ← selecionTorneo(Poblacion, 2);
7      Hijos ← ∅;
8      forall P_i, P_j ∈ Padres do
9          H_i, H_j ← cruzeDosPuntos(P_i, P_j, P_CRX);
10         Hijos ← Hijos ∪ mutacion(H_i, Red, P_MUT);
11         Hijos ← Hijos ∪ mutacion(H_j, Red, P_MUT);
12     Poblacion ← selNoDominada(Poblacion ∩ Hijos);
13     gen ← gen + 1;
```

Algoritmo 6: mutacion

Data: Ind, Red, P_MUT
Result: Ind

```
1  if rand() < P_MUT then
2      p ← 1/longitud(Ind);
3      forall i ∈ [0..longitud(Ind)] do
4          if rand() < p then
5              Ind[i] ← vecinoAzar(i, Red);
```

3.3.3. Batería de pruebas

Para realizar los experimentos se ha preparado una batería de pruebas compuesta por 66 redes sintéticas[1], cada una con 10 instantáneas, y dos redes reales con *ground truth*, cada una con 30 instantáneas.

Las redes dinámicas reales se generaron a partir de datos extraídos del juego online Travian [41]. En ambas redes, cada nodo representa un jugador y cada arista una interacción entre dos jugadores. Dependiendo del tipo de interacción, enviar un mensaje o comerciar, se genera una red diferente. Ambas redes contienen los datos de un mes entero, donde cada instantánea de la red contiene todas las interacciones que se llevaron a cabo en un día. Por último, el *ground truth* se extrajo a partir del gremio al que pertenecían los jugadores dentro del juego.

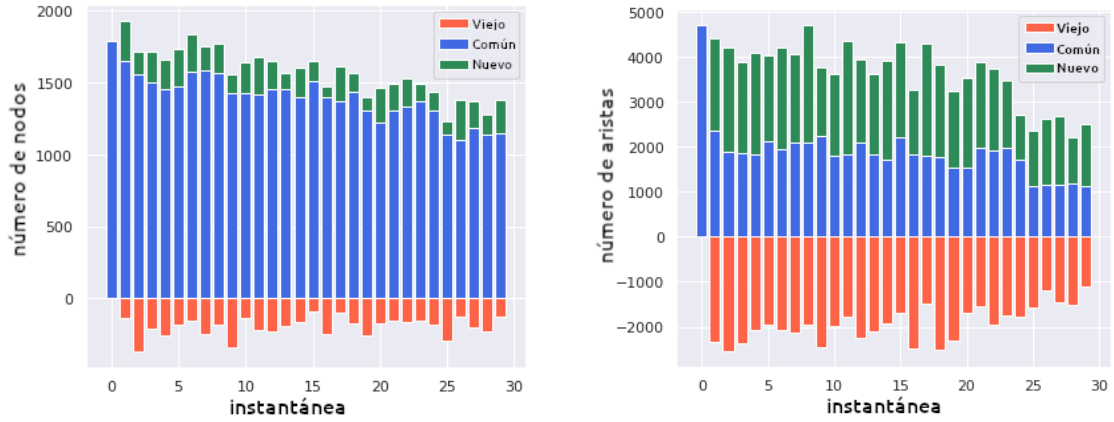
Respecto a las redes sintéticas, hemos utilizado el software DANCER [54], que es capaz de generar redes sintéticas dinámicas con estructuras de comunidad similares a las encontradas en el mundo real. Esta herramienta modela la dinámica de la red utilizando dos tipos de operaciones: las *micro operaciones*, que añaden o eliminan nodos y aristas, y las *macro operaciones*, que trabajan con comunidades enteras fusionándolas o dividiéndolas. Cada una de las 66 redes generadas utiliza una combinación diferente de micro/macro operaciones.

Obviamente, las redes que sufren menos cambios entre instantáneas son más fáciles de procesar y es necesario analizar la dificultad de cada una de las redes de la batería de pruebas, en aras de hacer una comparación justa. Las Figuras 3.6 y 3.7, muestran el número de nodos/aristas que cambian entre instantáneas consecutivas en las redes de mensajes y comercio de Travian. Tal cómo muestran las figuras, la red de comercio de Travian sufre muchos más cambios que la red de mensajes y por lo tanto será más difícil de analizar. En este caso se puede ver que el porcentaje de nodos/aristas nuevos (marcados en color verde) y eliminados (marcados en rojo) es mucho mayor en la red de comercio comparada con la red de mensajes.

Analizar las redes sintéticas de manera gráfica es inviable debido a su número y por eso se ha decidido hacerlo de manera cuantitativa mediante la métrica del *graph edit* [89]. Esta métrica cuenta el número de nodos y aristas que hay que modificar para que dos grafos sean iguales. Su valor mínimo es 0, si ambos grafos son isomorfos, mientras que su valor máximo es $n_a + e_a + n_b + e_b$, cuando ambos grafos son completamente distintos, donde en la formula anterior n_a y n_b indica el número de nodos del primer y segundo gráfico respectivamente, mientras que e_a y e_b indica el número de aristas. La Tablas 3.1 y 3.2 muestran el *graph edit* medio de cada una de las redes sintéticas de la batería de pruebas; nótese que cada red tiene tres niveles de dificultad que van del 1 al 3, siendo el 1 el más fácil y el 3 el más dificil. La primera tabla contine aquellas redes dónde el número de nodos es el mismo para todas las instantáneas de la red y la segunda las redes dónde el número de nodos cambia. Todas las redes de la batería están compuestas por un conjunto de micro/macro operaciones distinto, las descripciones de cada red se encuentran en la columna "Red" o "Descripción" de las tablas. Además del conjunto de operaciones aplicado, algunos casos permiten una mayor configuración que se ve reflejada en la columna "Variante" de las tablas.

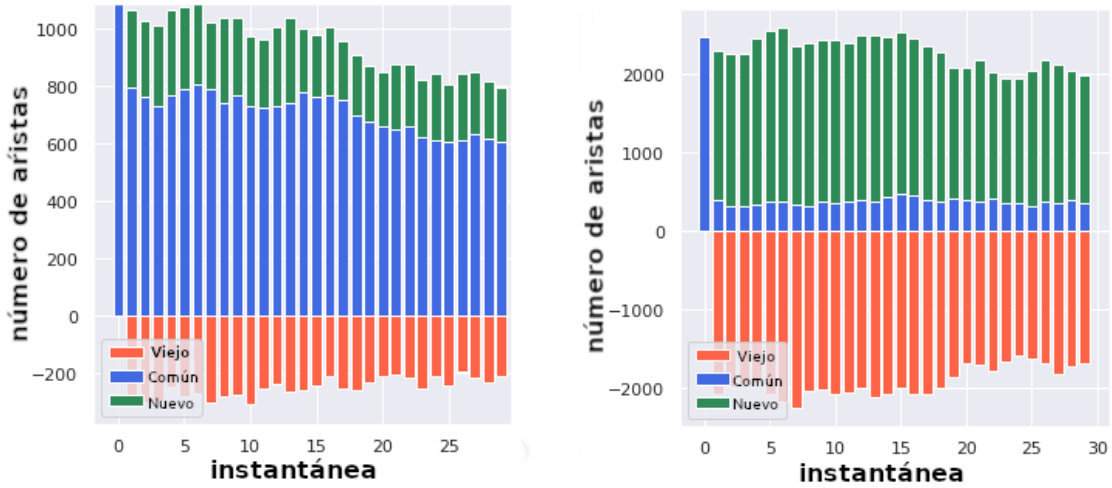
En los casos que trabajan con micro operaciones las variantes surgen por la manera de insertar/eliminar aristas, ya que puede afectar a la dificultad de encontrar una estructura de

[1]Las redes están disponibles para su descarga desde el siguiente enlace de Kaggle `https://www.kaggle.com/apanizolledot/dancer-synthetic-dynamic-networks`



(a) (b)

Figura 3.6: *Diferencia de nodos y aristas entre dos instantáneas consecutivas de red de Mensajes en Travian. El eje Y muestra la diferencia de nodos/aristas, el X el día del mes en que se tomó la instantanea y el color de cada barra indica en número de nodos/aristas nuevo (verde), eliminado (rojo) o que permanecen (azul) respecto a la instantánea anterior.*



(a) (b)

Figura 3.7: *Diferencia de nodos y aristas entre dos instantáneas consecutivas de red de Comercio en Travian. El eje Y muestra la diferencia de nodos/aristas, el X el día del mes en que se tomó la instantanea y el color de cada barra indica en número de nodos/aristas nuevo (verde), eliminado (rojo) o que permanecen (azul) respecto a la instantánea anterior.*

Red	**Variante**	**Graph Edit**		
		01	02	03
	Sólo micro operaciones			
fn_ge: Sólo se generan aristas nuevas entre instantáneas	**c+**: se añaden más aristas intracomunitarias	36	75	128
	d+: se añaden más aristas extracomunitarias	30	64	75
	eq: las aristas se añaden de manera homogenea	111	121	233
fn_le: Sólo se eliminan aristas entre instantáneas	**c+**: se eliminan más aristas extracomunitarias	34	64	87
	d+: se eliminan más aristas intracomunitarias	47	54	89
	eq: las aristas se eliminan de manera homogénea	78	110	149
fn_mx: Se eliminan y añaden aristas entre instantáneas	**c+**: Se añaden más aristas dentro de las comunidades que entre ellas	87	265	476
	d+: Se añaden más aristas entre comunidades que dentro de ellas	71	218	437
	eq: las aristas se repaten de manera homogéna	169	319	390
	Micro y macro operaciones			
fn_sp: Las comunidades sólo se dividen	NA	27	49	03
fn_mg: Las comunidades sólo se fusionan	NA	31	31	51
fn_sm: las comunidades se dividen y fusionan	**eq**: Las fusiones y divisiones se producen con la misma probabilidad	62	75	100
	m+: Las fusiones se producen con más frecuencia	25	52	86
	s+: Las divisiones se producen con más frecuencia	52	39	118

Tabla 3.1: *Nomenclatura seguida en las redes dinámicas sintéticas con un número fijo de nodos.*

Red	**Descripción**	**Graph Edit** 01	02	03
	Sólo micro operaciones			
ln	Sólo se eliminan nodos entre instantáneas	80	90	181
gn	Sólo se añaden nodos entre instantáneas	313	743	1249
mx	Se eliminan y se añaden nodos entre instantáneas	429	1014	1328
	Micro y macro operaciones			
sp	Las comunidades sólo se dividen	990	1120	1267
mg	Las comunidades sólo se fusionan	929	1149	1203
Red	**Variante**	**Graph Edit** 01	02	03
sm: Las comunidades se dividen y se fusionan	**eq**: Las fusiones y divisiones se producen con la misma probabilidad	1159	1203	1255
	m+: Las fusiones son más frecuentes que las divisiones	1054	1300	1157
	s+: Las divisiones son más frecuentes que las fusiones	943	1195	1390

Tabla 3.2: *Nomenclatura seguida en las redes dinámicas sintéticas con un número variable de nodos.*

comunidad concreta. Por ejemplo, si eliminamos más aristas que unen miembros de una misma comunidad que aristas que unen miembros de comunidades distintas provocaremos que las comunidades sean cada vez más difíciles de detectar. Por esta razón para las redes que incluyen micro operaciones se creó una variante dónde las comunidades tienden a ser más fáciles de detectar con el tiempo (c+), otra dónde las comunidades son más difíciles de detectar en el tiempo (d+) y, por último, una dónde no hay una tendencia clara (eq).

De la misma manera al trabajar con macro operaciones y aplicar fusiones y divisiones a la vez sobre la misma red, se puede dar la dicotomía de que una operación se aplique con más frecuencia que otra, para esos casos se han creado variantes dónde las fusiones se aplican con más frecuencia (m+), dónde las divisiones se aplican con más frecuencia (s+) o dónde no hay difirencia la frecuencia de ambas operaciones (eq).

Siguiendo la nomenclatura explicada en el párrafo anterior, el nombre de una red indica las operaciones que la conforman. Por ejemplo, la red *sm_eq_03* está compuesta por macro operaciones de fusión y división dónde no hay diferencia en la frecuencia de aparición de ambas y es la más difícil de analizar de sus hermanas, ya que es la red con un graph edit mayor. Por poner otro ejemplo, la red *mx_01* está compuesta solo de micro operaciones dónde tanto se eliminan como se añaden nodos y aristas entre instantáneas y es la más fácil de analizar de sus tres hermanas al tener el graph edit medio más bajo.

3.3.4. Experimentación

Para tener una referencia común con la que poderse comparar durante toda la experimentación se utilizará el MOGAs descrito en la sección 3.3.2 sin la reparación de poblaciones, llamado a partir de ahora el *Estándar*. Nótese, que el Estándar, al carecer de mecanismos de reparación, crea una población aleatoria completamente nueva cada vez que aparece un cambio en la red. La Tabla 3.3 muestra los parámetros selecionados para los dos MOGA propuestos para este caso de uso, el Estándard y el que repara poblaciones. Estos parámetros se han elegido a mano de forma experimental. Por último, ambos algortimos han sido desarrollados en Python 3.6, utilizando la implementación de NSGA-II disponible en la librería DEAP[2]. Asimismo, toda la experimentación de este trabajo se ha realizado en un ordenador Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20GHz con 125 GB de RAM.

Debido a la naturaleza estocástica de los algoritmos genéticos, cada experimento se ha repetido 20 veces y, para mayor robustez de los resultado, se han realizados test estadísticos. En concreto, los resultados de los experimentos se han comparado utilizando la prueba de Kruskall-Wallis [50] con un nivel de significancia de 0.05. Por último, para facilitar la reproducibilidad de la experimentación aquí presentada, todo el código necesario para llevarla a cabo está disponible en un repositorio público de github[3].

3.3.4.1. Eficiencia de los operadores de reparación

El primer experimento en el caso de uso pretende evaluar la eficiencia de los operadores de reparación para discernir si hay alguno mejor. Para ello, se repararán una serie de poblaciones

[2] `https://deap.readthedocs.io/en/master/`

[3] `https://github.com/panizolledotangel/Dynamic-communitie-Immigrants-MOGA`

Parámetros del MOGA	
Tamaño del torneo	2
Tamaño de la población	300
Tamaño de la Descendencia	300
Probabilidad de Cruce (P_CRX	100 %
Probabilidad de Mutacion (P_MUT)	10 %
Generaciones sin cambios (N)	20
Generaciones máximas (MAX_GEN)	300
Parámetros de la reparación	
Número de caminantes ($num_caminantes$)	6
Número de pasos (num_pasos)	5

Tabla 3.3: *parámetros del MOGA*

usando cada operador y se compararán los HVs obtenidos en cada caso. El método que genere poblaciones con un HV mayor será considerado el mejor, puesto que las soluciones generadas estarán mejor distribuidas en el espacio de búsqueda. Las poblaciones reparadas en este experimento serán aquellas generadas por el Estándar, tras el proceso evolutivo llevado a cabo para cada instantánea. Por clarificar, son las poblaciones que pasan del Bloque NSGA-II al bloque de Reparar población en la Figura 3.5. Téngase en cuenta que en este experimento solo se utilizaron las poblaciones generadas para los casos sintéticos de la batería de pruebas.

Las Tablas 3.4 y 3.5 muestran el hipervolumen medio obtenido por cada operador de reparación en las redes donde el número de nodos es fijo y el número de nodos es variable, respectivamente. Cada celda de la tabla muestra la media y la desviación estándar del HV obtenido en las 20 repeticiones de cada experimento. Cuando una fila de la tabla se marca con una cruz (✗) indica que la prueba H de Kruskal-Wallis obtuvo un p_valor mayor de $0,05$ y por lo tanto, no hay suficiente evidencia estadística para afirma que existe una diferencia entre el rendimiento de los operadores. Por el contrario, cuando una fila se marca se marca con un tick (✓) indica que la prueba H de Kruskal-Wallis obtuvo un p_valor menor de $0,05$ y que, por lo tanto, hay suficiente evidencia estadística para afirma que existe una diferencia entre el rendimiento de los métodos. En estos casos, el método que da mejor rendimiento se muestra en negrita. En aras de la reproducibilidad, cabe destacar que los parámetros ($num_caminantes$ y num_pasos), utilizados en el operador basado en random walks, se han fijado en 6 y 5 respectivamente. Estos valores se han ajustado experimentalmente.

Al analizar la Tabla 3.4, podemos ver que en la mayoría de los casos todas las técnicas de reparación muestran un rendimiento similar y en los casos en los que sí hay diferencia, no existe una técnica que domine al resto. En algunas redes la técnica basada en Label Propagation funciona mejor que la basada en Random Walks, mientras que en otras pasa lo contrario. En cuanto al rendimiento de la técnica híbrida, podemos ver qué hay redes, como fn_mx_c+_02, donde esta técnica muestra mejores resultados que cualquiera de las otras dos técnicas por separado. Sin embargo, puede observarse que la diferencia de rendimiento de las tres estrategias es baja.

Por otro lado, analizando los resultados mostrados en la Tabla 3.5, se puede observar que para redes en las que el número de nodos varía entre instantáneas, la técnica basada en Random

Walks es mejor que la de Label Propagation. Además, el uso del método híbrido no ofrece un mejor rendimiento que usar únicamente el método de Random Walks. Por último, puede observarse que, en este caso, la diferencia de rendimiento entre los métodos es más notable.

Para analizar en profundidad este fenómeno, hemos estudiado el rendimiento de cada operador a nivel de instantánea en lugar de a nivel de toda la red dinámica. Estos estudios han demostrado que la diferencia de rendimiento, entre el método basado en Label Propagation y el basado en Random Walks, depende mayormente del número de nodos nuevos que hayan aparecido en una red respecto a la instantánea anterior. Un ejemplo claro de este fenómeno se puede ver en la red mx_03, porque durante su evolución tiene instantáneas en las que sólo se eliminan nodos, instantáneas en las que se añaden muchos nodos nuevos y se eliminan muy pocos e instantáneas en las que se dan ambos casos. La Figura 3.8 muestra la fluctuación del número de nodos entre las instantáneas de dicha red y los Frentes de Pareto obtenidos en las instantáneas 3, 4 y 2 de esta. En la subfigura (a), la parte roja de las columnas muestra el número de nodos que han desaparecido de la red; la parte verde muestra el número de nodos nuevos y la parte azul muestra el número de nodos que permanecen entre dos instantáneas consecutivas de la red. En las subfiguras (b), (c) y (d) los puntos rojos representan las soluciones devueltas por el método basado en Random Walks y los puntos verdes los resultados devueltos por el basado en Label Propagation. Fíjese que en la subfigura (c), los Frentes de Pareto obtenidos por ambos métodos son más o menos iguales; además si miramos la instantánea 3 de la subfigura (a) podemos ver que no se añaden nuevos nodos. Por el contrario, en la subfigura (b) podemos ver que las soluciones obtenidas por el método basado en Random Walks dominan por completo a las soluciones obtenidas por el método basado en Label Propagation. En este caso, si observamos la instantánea 2 en la subfigura (a), veremos que han surgido bastantes nodos nuevos (barra verde). Por último, podemos observar en la subfigura (d) cómo este efecto de dominancia del método basado en Random Walks sobre el basado en Label propagation se mitiga, y cómo en la instantánea 4 de la subfigura (a) el número de nodos nuevos es menor que en la dos. Aunque en este tomo sólo se muestre este caso como ejemplo, los resultados obtenidos para el resto de las redes son similares, por lo que podemos concluir que la técnica **Random Walks** es la mejor técnica de reparación. En cuanto a la técnica Híbrida, en nuestra opinión, el salto de complejidad que se necesita para ejecutarla (téngase en cuenta que hay que ejecutar ambos métodos Label Propagation y Random Walks) no justifica su uso a la vista de los resultados obtenidos.

3.3.4.2. Impacto de la reparación de poblaciones en el rendimiento del MOGA

Una vez escogido el mejor método para reparar poblaciones, se pasará a analizar cómo afecta dicha estrategia de reparación al rendimiento del MOGA. Es de esperar que el número de generaciones que necesite el MOGA para procesar una red dinámica disminuya y en esta sección se comprobará en qué medida; además, también se comprobará si esta técnica tiene algún impacto positivo o negativo inesperado sobre el rendimiento del MOGA. Para ello se han llevado a cabo dos experimentos usando un subconjunto de la batería de pruebas: (1) En el primero, se compara el número medio de generaciones que requiere el Estándar y el MOGA con reparación de poblaciones. (2) En el segundo se compara la calidad de las estructuras de comunidad obtenidas por ambos métodos usando el *ground truth* de las rede sintéticas y la métrica del NMI.

Para la experimentación de esta sección se ha utilizado un subconjunto de las redes sintéticas de la bateria de pruebas. En concreto, se han seleccionado aquellas donde se utilizaban todas las

Red	Label Propagation	Random Walks	Híbrido	
fn_le_c+_01	8928.19 ± 14.45	8929.44 ± 9.74	8931.44 ± 1.76	✗
fn_le_c+_02	8898.06 ± 12.95	8904.38 ± 1.88	**8914.48 ± 8.84**	✓
fn_le_c+_03	8884.17 ± 16.41	**8926.34 ± 8.06**	8927.21 ± 8.71	✓
fn_le_d+_01	8853.12 ± 2.57	8835.49 ± 22.15	8862.37 ± 21.28	✗
fn_le_d+_02	8852.50 ± 18.05	8838.32 ± 16.51	8858.04 ± 18.91	✗
fn_le_d+_03	**8835.88 ± 17.09**	8819.85 ± 11.90	8843.55 ± 13.47	✓
fn_le_eq_01	**8904.13 ± 14.30**	8893.76 ± 7.51	8911.72 ± 8.95	✓
fn_le_eq_02	**8905.63 ± 12.30**	8877.42 ± 15.83	891.52 ± 8.14	✓
fn_le_eq_03	879.22 ± 25.22	**8827.40 ± 14.59**	8837.00 ± 12.16	✓
fn_ge_c+_01	8931.46 ± 7.68	8931.46 ± 7.68	8931.46 ± 7.68	✗
fn_ge_c+_02	8887.66 ± 16.57	8887.66 ± 16.57	8887.66 ± 16.57	✗
fn_ge_c+_03	8906.78 ± 14.10	8906.78 ± 14.10	8906.78 ± 14.10	✗
fn_ge_d+_01	8909.25 ± 7.08	8909.25 ± 7.08	8909.25 ± 7.08	✗
fn_ge_d+_02	8868.06 ± 9.28	8868.06 ± 9.28	8868.06 ± 9.28	✗
fn_ge_d+_03	8882.65 ± 2.80	8882.65 ± 2.80	8882.65 ± 2.80	✗
fn_ge_eq_01	8905.63 ± 6.67	8905.63 ± 6.67	8905.63 ± 6.67	✗
fn_ge_eq_02	8889.74 ± 17.40	8889.74 ± 17.40	8889.74 ± 17.40	✗
fn_ge_eq_03	8887.93 ± 2.80	8887.93 ± 2.80	8887.93 ± 2.80	✗
fn_mx_c+_01	8912.94 ± 1.73	**8928.67 ± 6.78**	8928.73 ± 6.84	✓
fn_mx_c+_02	8772.25 ± 3.59	8783.04 ± 18.24	**8811.07 ± 18.21**	✓
fn_mx_c+_03	8672.74 ± 22.17	**8721.93 ± 32.09**	8742.70 ± 24.48	✓
fn_mx_d+_01	8849.42 ± 19.17	8842.91 ± 18.36	8856.44 ± 2.78	✗
fn_mx_d+_02	8768.42 ± 27.35	**8788.95 ± 21.31**	8801.78 ± 21.85	✓
fn_mx_d+_03	8642.81 ± 24.71	8664.92 ± 19.80	**8688.25 ± 14.66**	✓
fn_mx_eq_01	8837.75 ± 21.73	883.98 ± 18.80	8846.95 ± 18.23	✗
fn_mx_eq_02	8662.69 ± 24.76	869.72 ± 28.22	**8716.87 ± 21.36**	✓
fn_mx_eq_03	8732.17 ± 2.34	8737.04 ± 12.63	**8769.48 ± 18.45**	✓
fn_sp_01	8904.12 ± 6.37	8901.15 ± 6.51	8904.21 ± 6.22	✗
fn_sp_02	8941.17 ± 5.88	8939.07 ± 6.59	8941.20 ± 5.83	✗
fn_sp_03	**8923.77 ± 7.83**	8909.12 ± 8.75	8925.45 ± 7.48	✓
fn_mg_01	17697.48 ± 23.52	17697.48 ± 23.52	17697.48 ± 23.52	✗
fn_mg_02	17653.14 ± 15.27	17653.14 ± 15.27	17653.14 ± 15.27	✗
fn_mg_03	17697.72 ± 19.42	17697.72 ± 19.42	17697.72 ± 19.42	✗
fn_sm_eq_01	1773.95 ± 2.87	17729.22 ± 19.18	17733.68 ± 19.59	✗
fn_sm_eq_02	17632.09 ± 25.52	17631.75 ± 26.01	17634.38 ± 24.97	✗
fn_sm_eq_03	17728.30 ± 21.54	17725.12 ± 23.92	17729.48 ± 22.32	✗
fn_sm_m+_01	1769.75 ± 18.90	17692.89 ± 2.09	17693.87 ± 19.14	✗
fn_sm_m+_02	17638.63 ± 16.46	17636.95 ± 15.42	17639.04 ± 17.23	✗
fn_sm_m+_03	17606.98 ± 32.04	17606.67 ± 33.27	17613.11 ± 31.06	✗
fn_sm_s+_01	8928.07 ± 1.81	893.58 ± 9.95	8932.13 ± 8.93	✗
fn_sm_s+_02	8928.67 ± 7.50	8926.88 ± 9.54	8929.35 ± 7.41	✗
fn_sm_s+_03	8868.24 ± 13.86	8863.91 ± 16.72	8875.24 ± 11.89	✗

Tabla 3.4: *Hipervolumen obtenido por los operadores de reparación para redes sintéticas con un número fijo de nodos. Un tick indica que hay suficiente evidencia estadística para considerar los resultados los diferentes, mientras que una cruz indica lo contrario. Cada celda muestra el valor medio y la desviación estándar de las 20 repeticiones.*

Red	Label Propagation	Random Walks	Híbrido	
ln_01	15481.46 ± 17.16	15483.55 ± 14.51	15488.66 ± 16.06	✗
ln_02	14419.41 ± 22.94	14416.96 ± 2.96	14421.20 ± 2.41	✗
ln_03	13833.64 ± 19.30	1383.66 ± 22.36	13834.42 ± 19.56	✗
gn_01	12035.59 ± 29.88	**12514.34 ± 17.41**	12513.83 ± 14.39	✓
gn_02	14582.52 ± 38.76	**15663.27 ± 2.01**	15663.43 ± 2.81	✓
gn_03	19599.25 ± 58.52	**21299.40 ± 3.44**	21299.33 ± 31.43	✓
mx_01	17462.13 ± 24.67	**18024.79 ± 18.07**	18027.36 ± 2.02	✓
mx_02	20065.22 ± 69.32	**21281.98 ± 32.00**	21285.46 ± 29.58	✓
mx_03	22936.06 ± 58.21	**25013.87 ± 42.47**	25013.57 ± 43.35	✓
sp_01	12214.30 ± 35.83	**12815.43 ± 36.79**	12814.18 ± 41.23	✓
sp_02	11984.81 ± 37.01	**12485.71 ± 39.42**	12491.48 ± 37.24	✓
sp_03	1222.70 ± 48.15	**12956.91 ± 25.10**	12958.25 ± 32.96	✓
mg_01	11652.49 ± 43.49	**1247.17 ± 2.98**	12473.36 ± 23.95	✓
mg_02	13934.59 ± 32.24	**1529.95 ± 38.24**	15297.85 ± 34.03	✓
mg_03	11259.11 ± 37.45	**11934.35 ± 27.87**	11938.57 ± 25.61	✓
sm_eq_01	12349.25 ± 69.52	**13117.50 ± 21.85**	13121.10 ± 23.80	✓
sm_eq_02	1244.41 ± 54.46	**13144.34 ± 26.44**	13143.07 ± 31.32	✓
sm_eq_03	12841.29 ± 61.06	**13497.83 ± 53.04**	1350.51 ± 51.07	✓
sm_m+_01	12452.29 ± 4.35	**1309.11 ± 34.70**	13097.42 ± 34.68	✓
sm_m+_02	13168.50 ± 69.71	**14222.04 ± 54.70**	14223.15 ± 62.60	✓
sm_m+_03	11767.94 ± 4.08	**1237.63 ± 27.91**	12382.08 ± 28.14	✓
sm_s+_01	1178.35 ± 33.68	**12441.00 ± 29.73**	1244.00 ± 27.17	✓
sm_s+_02	11143.76 ± 35.95	**11737.89 ± 21.14**	11739.53 ± 21.51	✓
sm_s+_03	13282.28 ± 5.27	**14135.52 ± 4.05**	14133.48 ± 36.23	✓

Tabla 3.5: *Hipervolumen obtenido por los operadores de reparación para redes sintéticas con un número variable de nodos. Un tick indica que hay suficiente evidencia estadística para considerar los resultados los diferentes, mientras que una cruz indica lo contrario. Cada celda muestra el valor medio y la desviación estándar de las 20 repeticiones.*

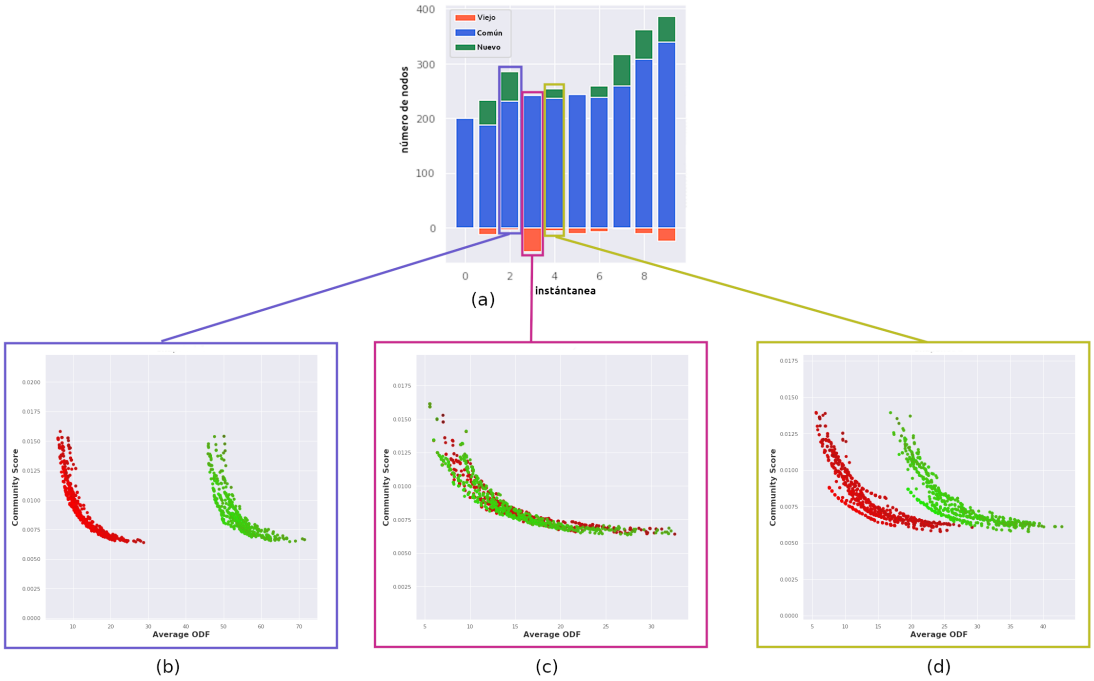


Figura 3.8: *Soluciones obtenidas para la red mx_03. La subfigura (a) contiene la diferencia de nodos entre dos instantáneas consecutivas y las subfiguras (b), (c) y (d) muestran el frente de Pareto obtenido para las instantáneas 2, 3 y 4 respectivamente. En los frentes de Pareto, los puntos verdes corresponden a los resultados del método basado en Label Propagation y los puntos rojos a los del método basado en Random Walks.*

Red	Estándar	Inmigrantes
fn_mx_eq_01	.1036 ± .0071	▲ .0851 ± .013
fn_mx_eq_02	.1990 ± .0144	≈ .1913 ± .0247
fn_mx_eq_03	.0550 ± .0083	▼ .0671 ± .0104
fn_sm_eq_01	.0899 ± .0068	≈ .0854 ± .0209
fn_sm_eq_02	.2565 ± .0094	≈ .2617 ± .0266
fn_sm_eq_03	.0908 ± .0064	≈ .0966 ± .0161
mx_01	.2900 ± .0140	▲ .2677 ± .0266
mx_02	.4034 ± .0139	≈ .4000 ± .0315
mx_03	.4466 ± .0104	▲ .4275 ± .0321
sm_eq_01	.1281 ± .0040	▲ .1168 ± .0153
sm_eq_02	.1632 ± .0120	≈ .1653 ± .0122
sm_eq_03	.1685 ± .0143	≈ .1640 ± .0165

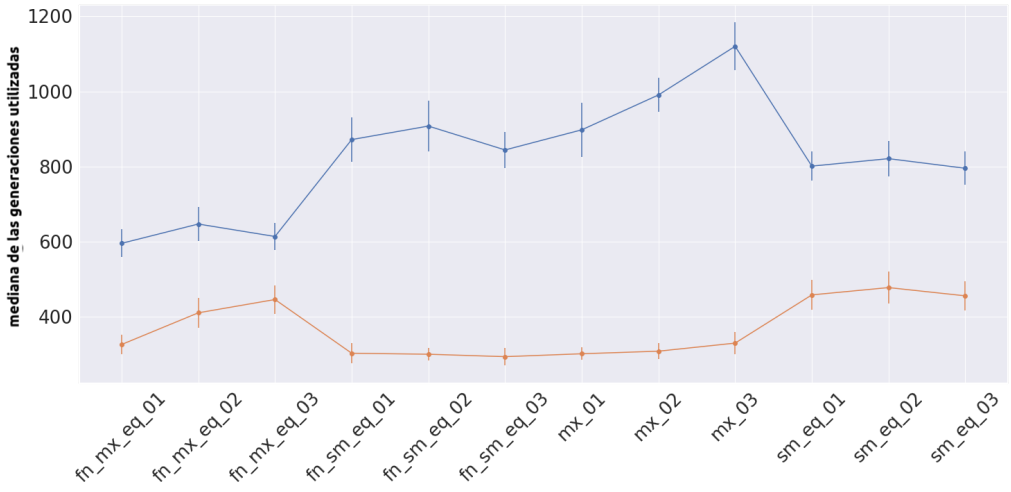
Tabla 3.6: *Error cuadrático medio (1-NMI). Cada celda muestra el valor medio y la desviación estándar de 20 repeticiones.*

micro y macro operaciones a la vez, ya que se asemejan más a las redes que podemos encontrar en el mundo real. Aislar las micro/macro operaciones por separado tenía sentido a la hora de analizar los operadores de reparación, ya que nos permitía encontrar sesgos. Sin embargo, ese grano fino no aporta nada en esta experimentación, por lo que se ha decido usar las redes más completas directamente.

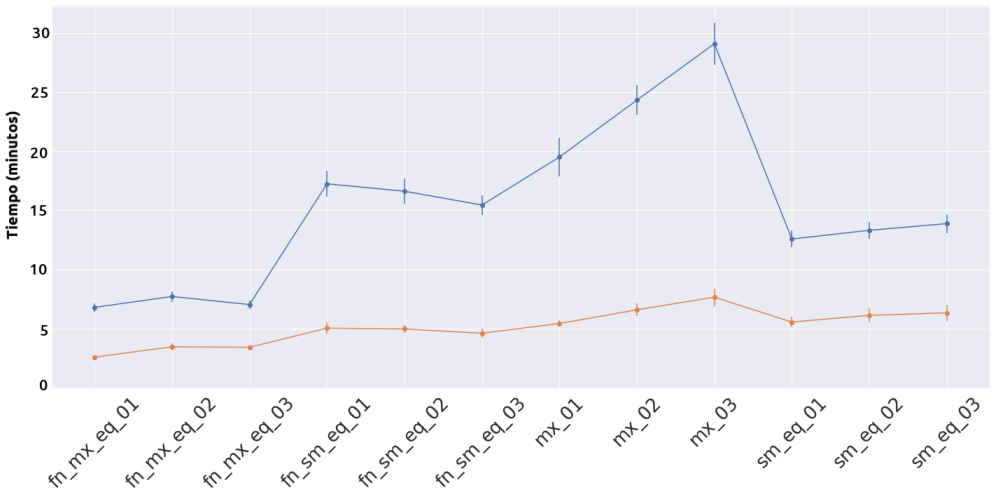
Comenzando con la reducción en el número de generaciones, la Figura 3.9 muestra el número medio de generaciones requeridas (arriba), y el tiempo consumido (abajo) para procesar todas las instantáneas de una red dinámica. La línea azul muestra los resultados del método Estándar y la línea naranja los resultados del método con reparación de poblaciones. Las líneas verticales muestran los valores mínimos y máximos obtenidos en las 20 repeticiones y el punto el valor medio. Como podemos ver, el uso de la reparación de poblaciones reduce drásticamente el número de generaciones y el tiempo necesario para procesar una red dinámica en todos los casos. La diferencia es tan pronunciada que el análisis visual es suficiente para asegurar que los resultados son significativos, sin la necesidad de llevar a cabo pruebas estadísticas. La mejora esperada al usar la reparación de poblaciones es de alrededor de 53 % de ahorro en tiempo de cómputo, con un pico máximo de 70 % y un mínimo de 30 %, lo que supone resultados muy prometedores, siempre y cuando la calidad de las soluciones no se vea afectada.

En cuanto a la calidad de las soluciones, la Tabla 3.6 muestra el error cuadrático medio de todas las instantáneas de una red obtenidas por cada método. El error de una instantánea se calcula como $1 - \text{NMI}$. Cada celda de la tabla muestra la mediana y la desviación estándar de las 20 repeticiones. Además, un símbolo ≈ indica que la prueba H de Kruskal-Wallis dio un *p_valor* superior a $0,05$, mientras que un triángulo indica lo contrario. Para los casos en los que sí existe una diferencia significativa, los marcados con un triángulo, si este es verde y apunta hacia arriba significa que el método mejora respecto al Estándar, mientras que si el triángulo es rojo y apunta hacia abajo significa que empeora.

Al analizar la tabla podemos ver que el MOGA con reparación de poblaciones encuentra soluciones de igual o mejor calidad que el Estándar para todas las redes excepto una. Téngase en cuenta, que cuanto menor es el error cuadrático medio mejor es la solución. Aunque es cierto



(a)



(b)

Figura 3.9: *Arriba, se muestra la mediana del de generaciones utilizadas y, abajo, el tiempo de ejecución. La línea azul es para el método Estándar y la naranja para el método con reparación de poblaciones. Las líneas verticales indican los valores mínimo y máximo obtenidos en las 20 repeticiones y el punto el valor medio.*

que la mejora es pequeña, tampoco es menos cierto que el objetivo principal de la reparación de poblaciones es reducir el esfuerzo computacional y no mejorar la calidad de los resultados. Sin embargo, es posible observar que la desviación estándar cuando usamos la reparación de población es mayor que cuando usamos el Estándar, lo cual no es un efecto positivo y debe mitigarse ya que hace el método más propenso a devolver soluciones de menor calidad y por lo tanto menos robusto. No obstante los resultados son positivos, ya que reparar las poblaciones reduce drásticamente el número de generaciones necesarias para procesar una red dinámica, sin afectar en gran medida a la calidad de la solución.

3.3.4.3. Comparación con el estado del arte

En el último paso de la experimentación se comparará el MOGA propuesto contra otros métodos del estado del arte. Para ello se usará el mismo subconjunto de redes sintéticas usado en la sección anterior y las dos redes reales. Se han escogido tres métodos del estado del arte contra los que hacer la comparación: DYN-MOGA [35], ALPA [42] e INFOMAP [87]. Dichos métodos se han escogido intentando abarcar la horquilla más amplia posible de aproximaciones al CDP dinámico.

- *DYN-MOGA*: se ha escogido por ser un algoritmo genético multi-objetivo como el nuestro. Las principales diferencias entre ambos algoritmos son dos: la primera, que DYN-MOGA sigue la metodología del evolutionary clustering mientras que nuestro método no tiene en cuenta la historia pasada de las comunidades, y la segunda, que nuestro método utiliza la reparación de poblaciones mientras que DYN-MOGA crea poblaciones nuevas cada vez que hay un cambio en la red.
- *ALPA*: se ha escogido por ser un método adaptativo capaz de actualizar la estructura de comunidad de manera eficiente analizando los cambios uno a uno, y, por tanto, es compatible con el modelo de redes temporales en vez de con el modelo de secuencia de instantáneas, que es el modelo compatible con nuestro algoritmo.
- *INFOMAP*: se ha seleccionado por ser un método clásico de detección de comunidades estáticas que, al igual que nuestro algoritmo, no tiene en cuenta la historia pasada de las comunidades.

Si analizamos los resultados mostrados en la Tabla **??** podemos ver que todos los métodos tienen un rendimiento más que aceptable en todas las redes; nótese que, salvo en la red *mx*, todas las demás tienen un error cuadrático medio por debajo de 0,1. En cuanto a cuál es mejor, sin duda DYN-MOGA tiene un mejor rendimiento que el resto en las redes con las que es compatible, lo que remarca la eficiencia del evolutionary clustering multi-objetivo para detectar redes dinámicas; sin embargo, para el resto de las redes los resultados son menos claros. En cinco casos INFOMAP muestra mejor rendimiento que el resto y en los tres casos restantes lo muestra ALPA, mientras que nuestro algoritmo no muestra mejor rendimiento en ninguno de los casos. Estos resultados parecen indicar que en redes simples como estas, dónde el índice de variación no es muy grande. no es tan importante tener en cuenta la historia de las comunidades, y, además, que en redes de tamaño pequeño como estas, la estrategia multiobjetivo no aporta más que otros métodos clásicos de búsqueda de comunidades. Nótese la mejor eficiencia de INFOMAP, que no tiene en cuenta la historia reciente de las comunidades, frente a ALPA, que sí lo tiene en cuenta,

Red	DYN-MOGA	ALPA	INFOMAP	Nuestro	
fn_mx_eq_01	**.0761 ± .0062**	.1555 ± 0	.0978 ± .0044	.0851 ± .0130	✓
fn_mx_eq_02	**.1551 ± .0127**	.1714 ± 0	.1660 ± .0049	.1913 ± .0247	✓
fn_mx_eq_03	**.0294 ± .0037**	.0849 ± 0	.0431 ± .0009	.0671 ± .0104	✓
fn_sm_eq_01	nan ± nan	**.0252 ± 0**	.0296 ± .0003	.0854 ± .0209	✓
fn_sm_eq_02	nan ± nan	.2089 ± 0	**.1812 ± .0025**	.2617 ± .0266	✓
fn_sm_eq_03	nan ± nan	**.0409 ± 0**	.0529 ± .0016	.0966 ± .0161	✓
mx_01	nan ± nan	.2126 ± 0	**.1881 ± .0024**	.2677 ± .0266	✓
mx_02	nan ± nan	**.2836 ± 0**	.3001 ± .0053	.4000 ± .0315	✓
mx_03	nan ± nan	.3800 ± 0	**.3507 ± .0046**	.4275 ± .0321	✓
sm_eq_01	nan ± nan	.0831 ± 0	**.0794 ± .0020**	.1168 ± .0153	✓
sm_eq_02	nan ± nan	.1292 ± 0	**.0746 ± .0030**	.1653 ± .0122	✓
sm_eq_03	nan ± nan	.1136 ± 0	**.0935 ± .0026**	.1640 ± .0165	✓

Tabla 3.7: *Error cuadrático medio (1-NMI). Cada celda muestra el valor medio y la desviación estándar de las 20 repeticiones. Una celda marcada con "nan ± nan" indica que un algoritmo no fue capaz de procesar la red correspondiente. Por último, un tick (✓) indica que hay suficiente evidencia estadística para considerar distinto el rendimiento entre los algoritmos, mientras que una cruz (✗) indicaría lo contrario*

y la mejor eficiencia de ambos métodos (mono-objetivo) frente al nuestro (multi-objetivo). Para finalizar, tal como se detectó en la sección anterior, nuestro método tiene una mayor desviación típica comparado con el resto de los métodos, lo cual es un problema que se debe revisar ya que hace el método menos robusto.

Respecto a las redes reales, las Tablas 3.8 y 3.9 muestran el error cuadrático medio obtenido por cada algoritmo para las dos redes sacadas del juego Travian y su desglose por instantánea. Téngase en cuenta que, en cada instantánea, se muestra el NMI y no el valor cuadrático medio, y por lo tanto, los valores hay que interpretarlos al contrario ya que cuanto más alto sea el valor mejor será el rendimiento.

Analizando la Tabla 3.8 se puede ver que los resultados obtenidos para las redes sintéticas se ven reforzados. En general, todos los métodos obtienen un rendimiento más que aceptable; véase que el error cuadrático medio está por debajo 0,05 para todos los algoritmos, y que, para las redes con pocos cambios, analizar la historia de las comunidades no aporta gran cosa (nótese que INFOMAP funciona mejor que el resto de los algoritmos en todas las instantáneas de la red de mensajes de Travian).

Sin embargo, si analizamos la Tabla 3.9 podemos ver cómo al aumentar el número de cambios, téngase en cuenta que la red de comercio cambia mucho más que la de mensajes, el rendimiento de los métodos disminuye notablemente: el error cuadrático medio se dispara y el NMI ha pasado de rondar 0,85 a rondar 0,55. Es curioso ver cómo INFOMAP obtiene sólo el mejor resultado en la primera instantánea mientras que nuestro algoritmo obtiene mejores resultados en el resto de ellas. Esto parece indicar que la reparación de poblaciones puede introducir cierta suavidad de en los resultados como efecto colateral. Ni INFOMAP ni nuestro MOGA tienen en cuenta la historia de las comunidades y a pesar de ello nuestro algoritmo muestra un rendimiento mayor comparado con él. Podría pensarse que este fenómeno es debido a la optimización multi-objetivo, pero el hecho de que INFOMAP sí muestre mejores resultados en la primera instantánea, en la

cual no hay información temporal que explotar, parece contradecir este hecho. Por lo tanto, no hay mejor explicación que suponer que la reparación de poblaciones introduce suavidad en las soluciones, aunque este fenómeno debe ser investigado en profundidad para obtener conclusiones más robustas.

Para finalizar, queda remarcar que los resultados tibios de ALPA se pueden deber al hecho de que este método se diseñó para trabajar con redes temporales y que al ser aplicado en redes dinámicas más propensas a ser tratadas como secuencias de instantáneas, puede haberle perjudicado. Este hecho deja de manifiesto que la elección del modelo de red temporal tiene implicaciones más allá de los datos en sí mismos.

3.4. Limitaciones y conclusiones

Tras la experimentación se puede concluir que el método de reparación basado en Random Walks es más eficiente que el basado en Label Propagation cuando aparecen nodos nuevos en una red dinámica, aunque su eficiencia es similar cuando el número de nodos decrece o se mantiene. Además, también podemos concluir que el operador de reparación híbrido no ofrece un rendimiento mejor a la aplicación que los otros dos reparadores por separado.

Respecto a los efectos que podemos esperar de aplicar la reparación de poblaciones a un MOGA, se ha comprobado que el número de generaciones y por tanto el tiempo de ejecución del algoritmo se reduce drásticamente sin verse afectada la calidad de los resultados. Los experimentos muestran que el tiempo de ejecución se redujo de media en un $\approx 50\,\%$ con picos máximos de un $\approx 70\,\%$ y mínimos de un $\approx 30\,\%$, mientras que la calidad se mantuvo igual o mejor en todos los casos estudiados menos uno. Sin embargo, se detectó un incremento en la desviación estándar del MOGA que usaba los operadores de reparación, lo cual es un efecto no deseado ya que hace de los algoritmos menos robustos, es decir, más propensos a devolver soluciones mediocres.

Por último, respecto a la comparación con el estado del arte, podemos concluir que los cuatro algoritmos probados (DYN-MOGA, ALPA, INFOMAP y el nuestro) obtienen resultados muy buenos en redes con una tasa de variación baja o moderada. En concreto, los experimentos sugieren que DYN-MOGA es el algoritmo más eficiente en aquellas redes con las que es compatible, mientras que para el resto INFOMAP presenta un rendimiento mejor que ALPA y a su vez ALPA un rendimiento mayor que nuestro algoritmo. Sin embargo, la jerarquía anterior no se mantiene en las redes con una tasa de variación alta. Los experimentos muestran una bajada significativa de la eficiencia en todos los algoritmos, pero menos pronunciada en el caso de nuestro método, y lo que es más sorprendente aún, que la reparación de las comunidades tiene como efecto colateral cierta suavidad en las soluciones, parecida a la buscada en la metodología del evolutionary clustering.

Aunque los resultados del caso de uso parecen prometedores, sería interesante aplicarlos en casuísticas más extensas que incluyeran otros tipos de MOGAs, como pueden ser MOEA/D o SPEA2, así como más redes dinámicas, en aras a probar más a fondo la eficiencia de los operadores de reparación. Además, es necesario atender al incremento de la desviación estándar detectado al aplicar las técnicas de reparación de poblaciones. La robustez de un algoritmo es una característica esencial si se quiere aplicar a problemas del mundo real y, en nuestra opinión, el estudio y desarrollo de técnicas que mitiguen este problema debería ser el próximo paso lógico a tratar.

Tabla 3.8: *NMI obtenidos en cada instantánea de la red de mensajes de Travian. Cada celda muestra el valor medio y la desviación estándar de las 20 repeticiones. Un tick (✓) indica que hay suficiente evidencia estadística para considerar distinto el rendimiento entre los algoritmos y una cruz (✗) indica lo contrario*

	Nuestro	**ALPA**	**INFOMAP**	
Error cuadrático medio	0.0233 +/- 0.0004	0.0110 +/- 0.0000	**0.0081 +/- 0.0000**	✓
Instantánea 0	.8351 ± .0058	.8805 ± .0000	**.9022 ± .0022**	✓
Instantánea 1	.8490 ± .0039	.8892 ± .0000	**.9155 ± .0006**	✓
Instantánea 2	.8253 ± .0046	.8681 ± .0000	**.8903 ± .0014**	✓
Instantánea 3	.8414 ± .0050	.8774 ± .0000	**.8885 ± .0009**	✓
Instantánea 4	.8440 ± .0045	.9003 ± .0000	**.9089 ± .0003**	✓
Instantánea 5	.8464 ± .0036	.9087 ± .0000	**.9210 ± .0010**	✓
Instantánea 6	.8467 ± .0031	.9005 ± .0000	**.9199 ± .0012**	✓
Instantánea 7	.8424 ± .0035	.8936 ± .0000	**.9104 ± .0014**	✓
Instantánea 8	.8478 ± .0040	.8970 ± .0000	**.9221 ± .0011**	✓
Instantánea 9	.8326 ± .0050	.8664 ± .0000	**.8912 ± .0021**	✓
Instantánea 10	.8365 ± .0044	.8857 ± .0000	**.8933 ± .0013**	✓
Instantánea 11	.8547 ± .0031	.9037 ± .0000	**.9335 ± .0009**	✓
Instantánea 12	.8402 ± .0042	.8975 ± .0000	**.9011 ± .0012**	✓
Instantánea 13	.8266 ± .0029	.8839 ± .0000	**.8982 ± .0009**	✓
Instantánea 14	.8527 ± .0040	.9123 ± .0000	**.9383 ± .0011**	✓
Instantánea 15	.8670 ± .0047	.9277 ± .0000	**.9451 ± .0016**	✓
Instantánea 16	.8366 ± .0041	.8978 ± .0000	**.9021 ± .0018**	✓
Instantánea 17	.8489 ± .0031	.9211 ± .0000	**.9371 ± .0011**	✓
Instantánea 18	.8573 ± .0030	.9086 ± .0000	**.9175 ± .0011**	✓
Instantánea 19	.8401 ± .0041	.8733 ± .0000	**.9063 ± .0019**	✓
Instantánea 20	.8468 ± .0049	.9212 ± .0000	**.9271 ± .0012**	✓
Instantánea 21	.8618 ± .0062	.9270 ± .0000	**.9341 ± .0015**	✓
Instantánea 22	.8752 ± .0045	.9144 ± .0000	**.9314 ± .0008**	✓
Instantánea 23	.8693 ± .0057	.9082 ± .0000	**.9292 ± .0016**	✓
Instantánea 24	.8725 ± .0025	.9058 ± .0000	**.9127 ± .0009**	✓
Instantánea 25	.8536 ± .0042	.8663 ± .0000	**.8809 ± .0012**	✓
Instantánea 26	.8510 ± .0037	.8883 ± .0000	**.9151 ± .0009**	✓
Instantánea 27	.8466 ± .0058	.9032 ± .0000	**.9052 ± .0017**	✓
Instantánea 28	.8397 ± .0040	**.8657 ± .0000**	**.8657** ± .0015	✓
Instantánea 29	.8708 ± .0032	.9053 ± .0000	**.9140 ± .0013**	✓

Tabla 3.9: *NMI obtenidos en cada instantánea de la red de comercio de Travian. Cada celda muestra el valor medio y la desviación estándar de las 20 repeticiones. Un tick (✓) indica que hay suficiente evidencia estadística para considerar distinto el rendimiento entre los algoritmos y una cruz (✗) indica lo contrario*

	Nuestro	**ALPA**	**INFOMAP**	
Error cuadrático medio	**0.1705 +/- 0.0009**	0.3620 +/- 0.0000	0.2130 +/- 0.0004	✓
Instantánea 0	.5249 ± .0103	.2507 ± .0000	**.5477 ± .0027**	✓
Instantánea 1	**.5932 ± .0037**	.5141 ± .0000	.5554 ± .0031	✓
Instantánea 2	**.6011 ± .0055**	.5265 ± .0000	.5482 ± .0035	✓
Instantánea 3	**.6057 ± .0039**	.4952 ± .0000	.5528 ± .0019	✓
Instantánea 4	**.5960 ± .0040**	.4999 ± .0000	.5497 ± .0040	✓
Instantánea 5	**.6008 ± .0036**	.3450 ± .0000	.5372 ± .0053	✓
Instantánea 6	**.5940 ± .0041**	.2827 ± .0000	.5299 ± .0021	✓
Instantánea 7	**.5904 ± .0040**	.4664 ± .0000	.5373 ± .0034	✓
Instantánea 8	**.5980 ± .0044**	.4690 ± .0000	.5441 ± .0034	✓
Instantánea 9	**.5896 ± .0033**	.4568 ± .0000	.5306 ± .0036	✓
Instantánea 10	**.5879 ± .0038**	.2888 ± .0000	.5317 ± .0029	✓
Instantánea 11	**.5935 ± .0054**	.5191 ± .0000	.5359 ± .0035	✓
Instantánea 12	**.5938 ± .0038**	.2668 ± .0000	.5389 ± .0055	✓
Instantánea 13	**.6073 ± .0034**	.5047 ± .0000	.5461 ± .0040	✓
Instantánea 14	**.6031 ± .0048**	.4558 ± .0000	.5468 ± .0039	✓
Instantánea 15	**.5861 ± .0042**	.2659 ± .0000	.5258 ± .0036	✓
Instantánea 16	**.5866 ± .0038**	.2386 ± .0000	.5242 ± .0039	✓
Instantánea 17	**.5956 ± .0044**	.4043 ± .0000	.5251 ± .0023	✓
Instantánea 18	**.5882 ± .0040**	.4509 ± .0000	.5345 ± .0036	✓
Instantánea 19	**.5820 ± .0048**	.4691 ± .0000	.5237 ± .0029	✓
Instantánea 20	**.5876 ± .0038**	.4677 ± .0000	.5186 ± .0040	✓
Instantánea 21	**.5881 ± .0056**	.3687 ± .0000	.5407 ± .0032	✓
Instantánea 22	**.5916 ± .0031**	.5002 ± .0000	.5441 ± .0053	✓
Instantánea 23	**.5916 ± .0063**	.5147 ± .0000	.5502 ± .0050	✓
Instantánea 24	**.6075 ± .0056**	.5074 ± .0000	.5529 ± .0042	✓
Instantánea 25	**.6020 ± .0057**	.2301 ± .0000	.5371 ± .0059	✓
Instantánea 26	**.5951 ± .0058**	.2591 ± .0000	.5431 ± .0050	✓
Instantánea 27	**.6066 ± .0045**	.4811 ± .0000	.5512 ± .0062	✓
Instantánea 28	**.5921 ± .0064**	.2711 ± .0000	.5272 ± .0041	✓
Instantánea 29	**.5925 ± .0044**	.4503 ± .0000	.5291 ± .0049	✓

Capítulo 4

Metodologías eficientes para adaptar poblaciones a cambios en la red

En este capítulo se presenta una metodología para evaluar estrategias de inmigrantes en algoritmos genéticos. Se comienza haciendo una introducción sobre las estrategias de inmigrantes, qué son, cómo funcionan y los desafíos que presenta su inclusión en un algoritmo genético (Sección 4.1). Una vez presentado el problema, se pasa a describir en detalle la metodología en sí (Sección 4.2), para a continuación, probar su validez con un caso de uso, en concreto, un MOGA para la detección de comunidades dinámicas (Sección 4.3). Por último, se exponen las conclusiones obtenidas tras el caso de uso, así como las limitaciones de este y futuras líneas de investigación (Sección 4.4). Todos los resultados expuestos en este capítulo están publicados en un articulo en la revista IEEE Access [78]; una versión de este artículo, libre del formato de la revista, está disponible en el anexo R3 de este tomo.

4.1. Introducción

Durante el proceso evolutivo llevado a cabo en un GA se mantiene un equilibrio entre dos fuerzas: i) una que tiende a *explorar* el espacio de búsqueda, es decir, buscar soluciones nuevas; ii) otra que tiende a *explotar* el espacio de búsqueda, para intentar mejorar una solución prometedora. Durante el proceso de evolución natural de una población es normal que se empieze realizando mayor esfuerzo de exploración que de explotación y, según se van encontrado soluciones intersante, se aumente la explotación entorno a esas soluciones a expensas de la exploración. Este proceso de convergencia es deseable ya que, sin él, un GA se volvería una búsqueda aleatoria. Sin embargo, éste mismo fenómeno se vuelve en nuestra contra al trabajar en entornos dinámicos. Si cuando llega un cambio al entorno la población del GA ya convergió, entonces, dicha población no será capaz de explorar el entorno de nuevo para encontrar soluciones prometedoras y se quedará estancada en una solución pasada que puede no ser la más deseable. Por esta razón, al trabajar en problemas dinámicos es necesario definir mecanismos extras que sean capaces de reequilibrar las fuerzas de explotación/exploración tras un cambio. Entre dichos me-

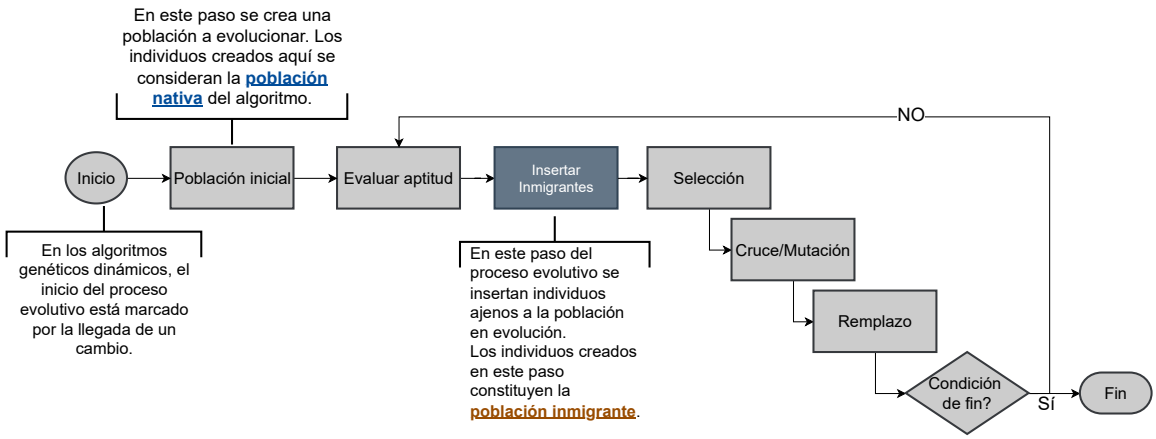


Figura 4.1: *Diagrama de secuencia de un algoritmo genético con una técnica de inmigrantes. Los bloques de color claro indican los pasos del algoritmo genético clásico y los de color oscuro los bloques añadidos por la técnica de inmigrantes.*

canismos, las estrategias de inmigrantes son una de las técnicas más populares [24, 107, 110, 22]. En gran medida, deben su popularidad a lo sencillo que es integrarlas en un algoritmo genético concreto. Sin embargo, ajustar una técnica de inmigrantes para obtener un rendimiento óptimo no es una tarea tan sencilla, ya que se tienen que establecer una serie de parámetros que pueden afectar de manera crítica al rendimiento de esta, como por ejemplo el número de inmigrantes que se van a insertar, la estrategia de reemplazo de esos inmigrantes o potenciar su probabilidad de supervivencia [117]. En aras de hacer dicho trabajo más sencillo, este capítulo propone una metodología que permite analizar en profundidad la eficacia de la aplicación de estos métodos para resolver problemas dinámicos y que, a su vez, dota al investigador de herramientas que lo ayuden a obtener información sobre lo que podría estar funcionando mal, en el caso que fuera necesario.

En pocas palabras, cualquier técnica de inmigrantes se basa en la idea de insertar nuevos individuos ajenos a la población actual en evolución para reemplazar una parte predefinida de esta. La hipótesis subyacente consiste en suponer que la inserción de nuevos individuos ayudará a mejorar la velocidad de convergencia y/o aumentar la diversidad de la población (dos de las características básicas en cualquier algoritmo evolutivo). La Figura 4.1 muestra el diagrama de flujo de un algoritmo genético que usa una técnica de inmigrantes. Los bloques de color claro marcan las operaciones de un genético clásico y los de color oscuro los que añade la técnica de inmigrantes. Tal como muestra la figura, se pueden identificar dos poblaciones distintas en el algoritmo: la población nativa y la población inmigrante. La primera está compuesta por los individuos creados de manera natural por el algoritmo genético antes del proceso evolutivo y la segunda está compuesta por individuos externos a la población nativa que se generan durante el proceso evolutivo y se insertan en esta. Nótese que distintas estrategias de inmigrantes utilizarán distintos tipos de individuos en cada población. Por ejemplo, RIGA utiliza una población reparada como la población nativa e individuos aleatorios como población inmigrante, mientras que EIGA utiliza individuos aleatorios como población nativa e individuos reparados de la instantánea anterior como población inmigrante. Conocer que ocurre entre estas poblaciones durante el proceso evolutivo es esencial para identificar las causas subyacentes que hacen de una estrategia de inmigrantes un éxito o un fracaso; y para abordar dicho problema se propone la metodología que se presenta a continuación.

4.2. Metodología para la evaluación de estrategias de inmigrantes

El objetivo de la metodología propuesta es triple. Primero permite comprobar si una estrategia de inmigrantes se está aplicando de manera exitosa. Segundo permite comparar estrategias de inmigrantes entre sí para saber cuál funciona mejor. Tercero y último, en caso de que una estrategia no obtenga los resultados esperados permite obtener información sobre las causas. Para poder considerar una estrategia de inmigrantes como exitosa, ésta debe converger más rápido que su contrapartida sin inmigrantes, es decir, ser más veloz. Además, las soluciones que encuentre tienen que tener una calidad igual o superior a las encontradas por el método sin inmigrantes y aparecer con la misma frecuencia; o sea, tener una estabilidad similar. Por esta razón, la metodología propuesta para evaluar el rendimiento de las estrategias de inmigrantes se basa en el empleo de tres métricas o dimensiones diferentes: *Calidad*, *Velocidad* y *Estabilidad*. En la siguiente sección se pasa a describir cada una de estas métricas en detalle, pero antes quisiera remarcar algunas consideraciones previas a tener en cuenta antes de aplicar la metodología.

Para poder aplicar la metodología propuesta en este capítulo es necesario cumplir algunos prerequisitos. Primero, presuponemos que hay disponibles N instancias diferentes del mismo problema; por ejemplo, N redes dinámicas si trabajamos en el CDP dinámico o N configuraciones de trabajos/máquinas si se trata del problema de planificación de trabajos (Job Scheduling Problem en inglés). Además, para lidiar con la naturaleza estocástica de los algoritmos genéticos, presuponemos que cada experimento (aplicación de una estrategia de inmigrantes concreta a una instancia del problema concreta) se ha repetido M veces. Téngase en cuenta que los valores de N y M tienen que ser elegidos teniendo en consideración el tipo de problema y la significación estadística deseada. Por último, damos por sentado que los cambios que se producen durante cada una de las N instancias del problema son conocidos a priori; por ejemplo, en el CDP, un cambio se refiere a variaciones en la topología de la red, es decir, la aparición/desaparición de nodos o aristas. Por otro lado, en el problema de planificación de trabajos, un cambio sería la entrada de un trabajo nuevo, la llegada de una máquina nueva o la retirada de una debido a una avería.

4.2.1. Dimensiones a evaluar

Como ya se ha dicho, cada esquema de inmigrantes se evalúa utilizando tres dimensiones diferentes: *Calidad*, *Velocidad* y *Estabilidad*. Cada dimensión se calcula utilizando los resultados de las M ejecuciones de cada instancia del problema de forma independiente. La Figura 4.2 muestra el flujo de trabajo para calcular cada dimensión que serán detalladas a continuación.

- *Calidad*: Esta dimensión mide, valga la redundancia, la calidad de los Frentes de Pareto generados por una estrategia de inmigración. Tal cómo se contó en el capítulo 2 de este tomo, una de las métricas más populares para medir la calidad de los Frentes de Pareto, y la que se utiliza en nuestra metodología, es el HV. El HV obtenido en cada experimento no se pude comparar directamente y se debe establecer un punto de referencia común y normalizar cada Frente de Pareto, si se desea realizar una comparación justa. Esta normalización utiliza los valores máximos y mínimos encontrados por todas las estrategias de inmigrantes en todas las instancias del problema y en todas las ejecuciones. Para calcular la métrica de Calidad de una instancia en particular, primero se calcula la mediana del HV

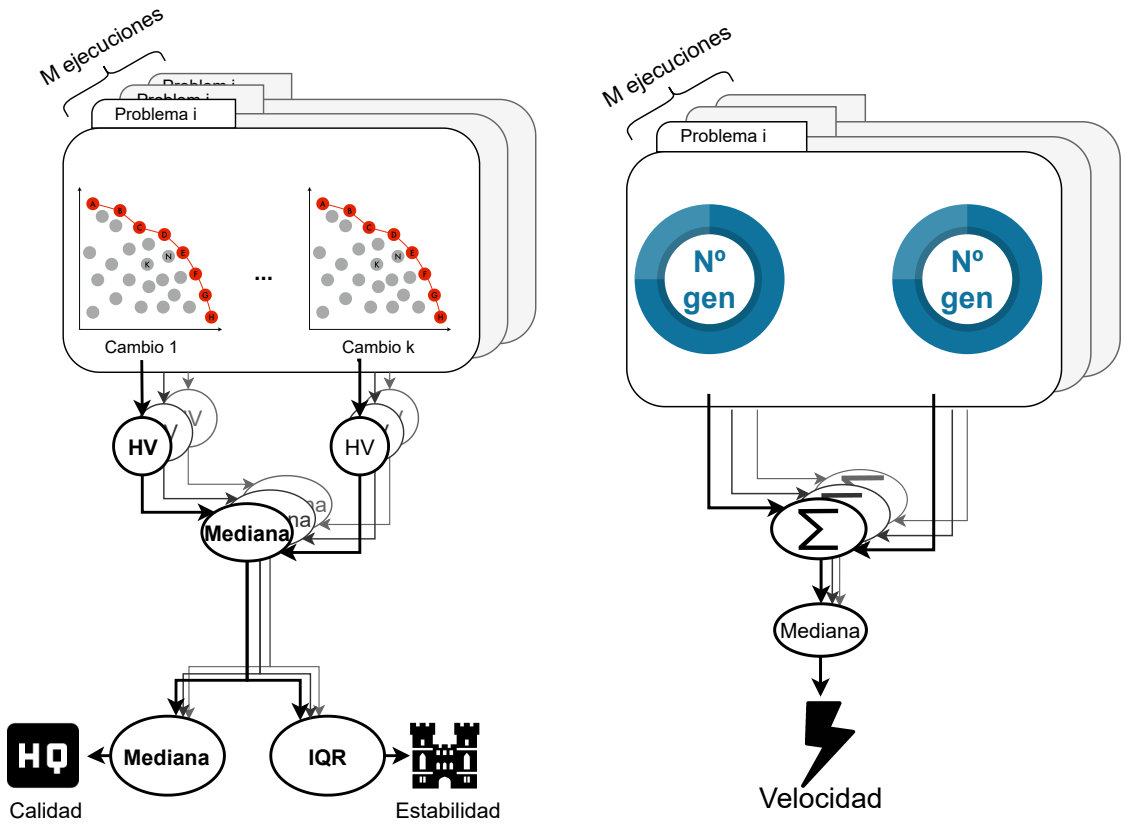


Figura 4.2: *a) Flujo de trabajo para evaluar las dimensiones Calidad y Estabilidad. b) Flujo de trabajo para evaluar la dimensión Velocidad.*

del Frente de Pareto normalizado generado después de cada cambio y para cada ejecución de la instancia del problema a tratar. A continuación, se utiliza la mediana de las M ejecuciones como valor final de la métrica *Calidad*. Cabe destacar que cuanto mayor sea el valor de esta métrica, mejor será el rendimiento de la estrategia de inmigración.

- *Estabilidad*: Esta dimensión mide la fiabilidad de los resultados de una estrategia de inmigrantes. Téngase en cuenta que dichas estrategias pueden influir en la capacidad de exploración de un MOGA, haciéndolo más o menos propenso a atascarse en mínimos locales. Cuando esto último ocurre, el MOGA se vuelve menos fiable, ya que la probabilidad de devolver soluciones mediocres aumenta. El procedimiento para calcular esta métrica es el mismo que el seguido en la métrica *Calidad*, excepto el último paso. Para la métrica *Estabilidad*, en lugar de utilizar la mediana de las M ejecuciones de cada problema, se utiliza el Rango Intercuartil de los valores, o IQR por sus siglas en inglés. El IQR es una medida de la dispersión de los datos. Por lo tanto, cuanto mayor sea el valor, menor será la estabilidad, provocando que la calidad de las soluciones obtenidas tenga una mayor variabilidad entre diferentes ejecuciones.

- *Velocidad*: Esta dimensión mide la velocidad de convergencia de una estrategia de inmigrantes. Se trata de un aspecto crucial, ya que el objetivo principal de las técnicas de inmigrantes es reutilizar la información previa (es decir, el conocimiento previo) para acelerar el proceso evolutivo. Para medir la métrica de velocidad en una instancia del problema, primero se calcula el número de generaciones necesarias para converger tras cada cambio, luego se suman dichos valores y, por último, se calcula la mediana de las M repeticiones de cada problema. Nótese que una estrategia de inmigrantes será más rápida que otra cuando se utilice un número menor de generaciones para converger y, por lo tanto, cuando menor sea este valor más rápido será el método.

4.2.2. Flujo de Trabajo

El flujo de trabajo de la metodología propuesta se compone de tres pasos. En primer lugar, se comparan todas las estrategias de inmigrantes bajo estudio contra el mismo MOGA pero sin inmigrantes. A partir de ahora, nos referiremos a este algoritmo como el *Estándar*. En el segundo paso, se comparan todas las estrategias de inmigrantes entre sí, con el fin de analizar cuáles presentan mejores resultados. Por último, se realiza un análisis de grano fino, en el que se evalúan instancias específicas del problema de forma independiente, utilizando la técnica de visualización de la *mezcla de poblaciones*. Esta técnica se ha desarrollado durante esta tesis expresamente para ser utiliza en la metodología. A continuación, se pasan a detallar cada uno de los pasos anteriores, para luego describir la técnica de visualización.

4.2.2.1. Paso uno: Comparación de los métodos bajo estudio con el estándar

En el primer paso, se comparan todas las estrategias de inmigrantes contra el *Estándar* para cada dimensión de manera independiente. Para ello, se realiza un test de Friedman con signo y un análisis post-hoc de Holm [29] sobre las puntuaciones obtenidas por cada estrategia de inmigrantes en todas las instancias del problema y así discernir si una estrategia inmigrante favorece o perjudica alguna dimensión. Para considerar una estrategia de inmigrantes exitosa

debe mejorar la dimensión *Velocidad* y mantener un rendimiento similar o mejor en las dimensiones de *Estabilidad* y *Calidad*. Dado que el objetivo principal de una estrategia de inmigrantes es reducir el tiempo de cómputo necesario cuando se utilizan MOGAs en entornos dinámicos, consideramos obligatoria la mejora de la dimensión *Velocidad*, mientras que en las otras dos dimensiones nos conformamos con que con que, al menos, funcione igual de bien que la técnica base o Estándar, es decir, la que no utiliza inmigrantes.

En cuanto a la visualización de los resultados, recomendamos presentar la información en una tabla en la que las filas contengan las diferentes estrategias de inmigración y las columnas las tres dimensiones. Además, cada celda de la tabla contendrá un triángulo verde apuntando hacia arriba (▲), si el puesto obtenido por la estrategia de inmigrante es mayor (menor en caso de minimización) que el obtenido por el *Estándar* y además hay suficiente evidencia estadística para apoyar que ambos métodos tiene un comportamiento diferente; un triángulo rojo apuntando hacia abajo (▼) en caso contrario, es decir, si el puesto obtenido es inferior (superior en caso de minimización); o, por último, el signo aproximado (≈) cuando no haya suficiente evidencia estadística para considerar que la estrategia del inmigrante tiene un comportamiento diferente al *Estándar*. Un ejemplo de la visualización propuesta se encuentra en la Tabla 4.4 (pág 69).

4.2.2.2. Paso dos: Comparación de los métodos bajo estudio entre sí

El paso anterior permite entender si una estrategia mejora o perjudica alguna dimensión, sin embargo, no da ninguna información sobre las diferencias entre los valores obtenidos por los algoritmos en cada dimensión. Por lo tanto, en aras de obtener dicha información, en el segundo paso, se comparan todos los resultados de las diferentes estrategias de inmigrantes entre sí utilizando una prueba de rango con signo de Friedman y un análisis post-hoc de Shaffer [29].

En cuanto a la visualización de este paso, se sugiere representar las relaciones entre las diferentes estrategias mediante un grafo, donde los nodos representen las diferentes estrategias de inmigración, su tamaño sea proporcional al rango obtenido y las aristas unan métodos cuando no haya suficiente evidencia estadística para considerar que tienen un rendimiento diferente. Nótese que, en algunas dimensiones, un rango mayor implica mejor rendimiento mientras que, en otras, implíca lo contrario; por eso, se sugiere que el tamaño de los nodos sea homogéneo en todos los grafos, de tal manera que nodos más grandes siempre implique un rendimiento mejor. Por último, un ejemplo de dicha visualización está disponible en la Figura 4.3 (pág 71).

4.2.2.3. Paso tres: Análisis de grano fino para identificar anomalías

Para aquellas estrategias que no funcionan como se esperaba, la metodología incluye este tercer paso, en el que se realiza un análisis de grano fino. En primer lugar, se identifican aquellas instancias del problema que presentan dificultades siguiendo el mismo procedimiento que en el descrito en el primer paso, salvo que cada instancia del problema se evalúa de forma independiente. Nótese que, en este paso, en lugar de utilizar la mediana de las M ejecuciones, se utilizan los M valores en sí. Una vez identificadas las instancias del problema que presentan dificultades, hay que identificar los periodos de tiempo donde el proceso de evolución no está funcionando correctamente. Para ello, se utiliza la evolución del HV(Figura 4.4 (b) pág 72) y sus Emperical Attainment Surfaces (Figura 4.4 (c) pág 72). Por un lado, la evolución del HV permite identificar si una determinada estrategia está sufriendo una convergencia prematura, mientras que,

por otro lado, las Emperical Attainment Surfaces permiten identificar si alguna área del Frente de Pareto no están siendo suficientemente explorada. Por último, una vez identificados dichos periodos, se aplica la técnica de visualización de la mezcla de poblaciones en ellos para obtener información sobre lo que puede estar obstaculizando el proceso de evolución. Esta técnica ayuda a discernir si los inmigrantes de una estrategia están contribuyendo a la mejora de la población, se están apoderando de ella o, por el contrario, están siendo excluidos de esta.

4.2.3. Técnica de visualización de la mezcla de poblaciones

Esta sección describe la técnica de visualización diseñada para analizar lo que ocurre con los inmigrantes durante el proceso de evolución. Para ello, cada individuo de la población se etiqueta con un tipo determinado. Los individuos iniciales se etiquetarán como nativos o inmigrantes y su progenie recibirá una etiqueta distinta dependiendo de las etiquetas de sus progenitores. Durante el proceso evolutivo, se contará el número de individuos de cada tipo en la población para cada generación y se representarán dichos valores en forma de curvas en un gráfico dónde el eje de las X contenga la generación (primera, segunda, tercera, etc.), mientras que el otro eje contenga el número de individuos en la población con cada tipo de etiqueta. La Figura 4.7 (pág 76) muestra varios ejemplos de este tipo de visualización donde cada curva es representada con un color diferente.

Los tipos de etiquetas disponibles para un individuo son:

- *n (nativo)*: individuo de la población original del algoritmo genético.
- *i (inmigrante)*: individuos externos a la población original, que se insertan continuamente durante el proceso evolutivo.
- *i+i (descendiente de inmigrantes)*: individuos cuyos padres son dos inmigrantes o descendientes de inmigrantes.
- *n+n (descendientes de nativos)*: individuos cuyos padres son dos nativos o descendientes de nativos.
- *n+i (mestizo)*: individuo que tiene un progenitor nativo o descendiente de nativos y el otro inmigrante o descendiente de inmigrantes.

La Tabla 4.1 muestra los cinco tipos de inmigrantes propuestos y el tipo de su progenie. Por ejemplo, cuando dos progenitores de tipo *'i'* se aparean (primera fila y columna de la tabla), la descendencia resultante será de tipo *'i+i'*. Del mismo modo, si, por ejemplo, los progenitores tienen los tipos *'n+n'* y *'n+i'* (tercera fila y cuarta columna de la tabla), la descendencia resultante será del tipo *'n+n'*.

Analizando las curvas de la mezcla de poblaciones es sencillo comprobar si la población inmigrante está contribuyendo al proceso de evolución, está conquistando a la población local o, si, por el contrario, está siendo excluida de ella. Para que una técnica de inmigrantes tenga éxito, es necesario que las dos poblaciones, la inmigrante y la nativa, cooperen entre sí, lo que se traduce en la presencia de individuos mestizos ($n+i$) entre las curvas de poblaciones. Si esto no pasa es muy posible que la técnica de inmigrantes en cuestión muestre una perdida de rendimiento.

	i	**i+i**	**n+i**	**n+n**	**n**
i	i+i	i+i	i+i	n+i	n+i
i+i		i+i	i+i	n+i	n+i
n+i			n+i	n+n	n+n
n+n				n+n	n+n
n					n+n

Tabla 4.1: *Las filas y columnas muestran el tipo de los padres, mientras que las celdas representan el tipo de su progenie. Como la tabla es simétrica, sólo se muestra la diagonal superior de la tabla.*

Incidiendo más en las causas de este fenómeno, nótese que cuando las curvas muestran una ausencia de mestizos y un exceso de descendientes de inmigrantes (i+i), podemos concluir que la población inmigrante es demasiado agresiva y está conquistando a la población nativa. Cuando este efecto es muy pronunciado, puede darse el caso de que toda la población nativa llegue a ser expulsada del proceso evolutivo antes de haber tenido oportunidad de transmitir sus genes a las siguientes generaciones, lo que sería equivalente a evolucionar la población inmigrante en solitario (con la pérdida de capacidad de exploración que ello conlleva). Si, por el contrario, las curvas muestran un exceso de descendientes de nativos, entonces podemos concluir que la población nativa es muy agresiva y está expulsando a la población inmigrante. Si este efecto es muy pronunciado, se puede llegar al extremo en el que todos los inmigrantes son expulsados de la población sin haber tenido la oportunidad de transmitir sus genes a las siguientes generaciones, lo que es equivalente a utilizar un tamaño de población menor, ya que la población inmigrante ocupa una parte de la población, pero no participa en el proceso evolutivo.

4.3. Caso de uso: MOGA aplicados al problema de CDP dinámicas

Para comprobar la validez de la metodología propuesta, se llevará a cabo un caso de uso que consiste en aplicar dicha metodología para evaluar qué esquema de inmigrantes funciona mejor con el MOGA para detectar comunidades dinámicas que ya se propuso en el capítulo 3 de este tomo. Primero se describirán el problema y las tecnicas de inmigrantes a analizar; luego, las redes dinámicas utilizadas como instancias del CDP en la metodología; a continuación, se detallará la configuración concreta del MOGA. Por último, se aplicará la metodología paso a paso, y se presentarán las conclusiones obtenidas durante este caso de uso.

4.3.1. Definición del problema

Dada una red dinámica $SN = \{G^0...G^T\}$ modelada como una secuencia de instántaneas, donde cada $G^t = (V^t, E^t)$ es un grafo que representa la red en un instante t de tiempo y está compuesto por un conjunto $V^t = \{v_1^t...v_n^t\}$ nodos y otro $E^t = \{e_{ij}^t | i \in V^t \wedge j \in V^t\}$ de aristas. El problema a resolver en este caso de uso consiste en encontrar una estructura de comunidad $CS^t = \{C_0^t...C_k^t\}$ para cada instántanea de la red G^t, donde cada $C_i^t = \{v_1^t...v_m^t | v_i^t \in V^t\}$ es un conjunto de nodos de la instántanea en cuestión. Trabajaremos con la versión del problema

no-solapada, por lo que cada nodo sólo podrá pertenecer a una única comunidad, es decir, $\forall t \in \{0...T\}, \bigcup_{C_i^t \in CS^t} C_i^t = \emptyset$.

4.3.2. Estrategias bajo estudio

En el caso de estudio se evaluarán tres de las estrategias de inmigrantes más comunes del estado del arte:

- RIGA [98]: Un pseudocódigo que implementa este método está disponible en el Algoritmo 7. En esta estrategia de inmigrantes, la población inicial está compuesta por las soluciones de la población anterior al cambio (línea 1). En el paso *Insertar Inmigrantes* (línea 11), se insertan en la población individuos aleatorios de nueva creación que sustituyen a los peores individuos. Esta técnica pretende mejorar la capacidad de exploración de un MOGA introduciendo diversidad en la población.

Algoritmo 7: RIGA

```
   Data: P^{t-1}, Red
   Result: P
1  P^t ← actualizarIndividuos(P^{t-1}, Red);
2  P^t ← evaluar(P^t, Red);
3  while !termination_criteria() do
4      padres ← seleccion(P^t);
5      hijos ← ∅;
6      forall p_1, p_2 ∈ padres do
7          Hs ← cruce&mutation(p_1, p_2, Red);
8          Hs ← evaluar(Hs, Red);
9          hijos ← hijos ∪ Hs;
10     P^t ← remplazo(P^t, hijos);
       //N individuos aleatorios remplazan a los peores de P^t
11     P^t ← insertarAleatorio(P^t, N);
```

- EIGA [109]: Un pseudocódigo que implementa este método está disponible en el Algoritmo 8. En esta estrategia de inmigrantes, la población inicial es una población aleatoria completamente nueva (línea 1). Además, la población del MOGA de antes de que se produjese el cambio se almacena en una memoria externa. En el paso *Insertar Inmigrantes* (línea 11), los individuos de dicha memoria externa sustituyen a los peores individuos de la población bajo evolución. Esta técnica pretende mejorar la velocidad de convergencia de un MOGA mediante la reutilización de información pasada.

- HIGA [111]: Un pseudocódigo que implementa este método está disponible en el Algoritmo 9. Esta estrategia de inmigrantes combina las dos técnicas anteriores. Dependiendo de las necesidades de la población, se insertan individuos aleatorios o de la población anterior al cambio. En este esquema, la población inicial es una población aleatoria completamente nueva (línea 1). Además, este método también almacena la población de antes del cambio

Algoritmo 8: EIGA

```
   Data: P^{t-1}, Red, POP_SIZE
   Result: P
 1 P^t ← poblacionAleatoria(POP_SIZE, Red);
 2 P^t ← evaluar(P^t, Red);
 3 while !termination_criteria() do
 4     padres ← seleccion(P^t);
 5     hijos ← ∅;
 6     forall p_1, p_2 ∈ padres do
 7         Hs ← cruce&mutation(p_1, p_2, Red);
 8         Hs ← evaluar(Hs, Red);
 9         hijos ← hijos ∪ Hs;
10     P^t ← remplazo(P^t, hijos);
       //N individuos de la población anterior remplazan a los peores de P^t
11     P^t ← insertarPoblacionAux(P^t, P_{t-1}, N);
```

en una memoria externa, al igual que EIGA. Sin embargo, este método combina la inserción de inmigrantes con la fase *cruce/mutación*. El método funciona como sigue; si ambos padres no son miembros del conjunto de soluciones no dominadas, se inserta un individuo de la memoria externa (líneas 7-8); si al menos uno de los padres es miembro del conjunto no dominado y la distancia entre ambos padres es inferior al 10 % de los genes, entonces se inserta un individuo completamente aleatorio (líneas 10-11); en caso contrario, se realiza el proceso de cruce y mutación clásico (líneas 12-13).

Además, se han incluido en la experimentación dos métodos adicionales que serán utilizados como puntos de referencia durante las comparaciones. Estos métodos son:

- Hypermutation Genetic Algorithm (Algoritmo genético de hipermutación) (HGA) [49]. Este método no se basa en una estrategia de inmigrantes per se, ya que, en lugar de insertar continuamente inmigrantes en la población, aumenta la tasa de mutación cuando se produce un cambio y continua el proceso evolutivo normal. Este método se incluye en la experimentación por ser más sencillo y requerir menos memoria que las estrategias de inmigrantes ya que no necesita almacenar ninguna población externa.

- Delayed Elite Immigrants Genetic Algorithms (Algoritmo genético con inmigrantes elitistas desfasado) (DEIGA): Este método sigue una estrategia de inmigrantes muy similar a la usada en EIGA, salvo por la diferencia que cuenta con un desfase a la hora de insertar los inmigrantes en la población nativa, es decir, se espera un número fijo de generaciones antes de comenzar a meter los inmigrantes en la población. Este método se incluye en el estudio ya que permite entender si las poblaciones inmigrante y nativa exploran diferentes secciones del espacio de búsqueda. Con este método podemos analizar si, dado el tiempo suficiente, tanto la población inmigrante como la nativa convergen a los mismos frentes de Pareto o a otros diferentes.

Algoritmo 9: HIGA

Data: P^{t-1}, Red, POP_SIZE
Result: P

```
1  P^t ← poblacionAleatoria(POP_SIZE, Red);
2  P^t ← evaluar(P^t, Red);
3  while !termination_criteria() do
4      padres ← seleccion(P^t);
5      hijos ← ∅;
6      forall p1, p2 ∈ padres do
7          if {p1, p2} ∩ setNoDominado(P^t) == ∅ then
               //Se necesita más explotación, insertar individuos de la
                 población anterior
8              Hs ← selIndividuos(P^{t-1});
9          else if distanciaGenetica(p1, p2) ≤ 0,1 then
               //Se necesita explorar, insertar individuo creado al azar
10             Hs ← individuosAleatorios(Red);
11         else
12             Hs ← cruce&mutation(p1, p2, Red);
13         Hs ← evaluar(Hs, Red);
14         hijos ← hijos ∪ Hs;
15     P^t ← remplazo(P^t, hijos);
```

4.3.3. Instancias del CDP dinámico y configuración del MOGA

Como instancias del CDP dinámico se han seleccionado 18 redes sintéticas del banco de pruebas presentado en el capítulo 3 de este documento (parámetro N en nuestra metodología) y se ha fijado en 20 el número de ejecuciones de cada MOGA (el parámetro M en nuestra metodología). Nótese, que las redes específicas que se han seleccionado se pueden encontrar en la Tabla 4.2.

Respecto a la configuración del MOGA, se usará la misma que la descrita en el capítulo 3 de este documento salvo por la condición de parada, aunque se sigue manteniendo el número máximo de generaciones en caso de que no se alcance la convergencia, que sigue siendo de 300 generaciones. Un resumen de los operadores y parámetros utilizados se pude encontrar en la Tabla 4.3. Para este caso de uso se utiliza como condición de parada una prueba de Chi-cuadrado de una sola muestra sobre el HV de los últimos 20 frentes de Pareto y un nivel de significación de 0,05. En relación a los parámetros añadidos por las técnicas de inmigrantes, siempre que no se diga lo contrario, se sustituye un 10 % de la población por inmigrantes. Para finalizar, el HV se calcula, primero invirtiendo el *Community Score* de las soluciones restándole $1,0$; después, normalizando los frentes de Pareto tal como se estipula en la sección 2 y, por último, calculando el HV utilizando $(1,1,1,1)$ como punto de referencia ideal.

Nombre	Descripción
mx_XX	Red que aplica *micro operaciones* durante su evolución con tres niveles de intensidad: baja (XX = 01), media (XX = 02) y alta (XX = 03)
mg_XX	Red que aplica la macrooperación de *fusión* durante su evolución con tres niveles de intensidad: baja (XX = 01), media (XX = 02) y alta (XX = 03)
sp_XX	Red que aplica la macrooperación de *división* durante su evolución con tres niveles de intensidad: baja (XX = 01), media (XX = 02) y alta (XX = 03)
sm_eq_XX	Red que aplica las macrooperaciones de *división* y de *fusión* a partes iguales durante su evolución con tres niveles de intensidad: baja (XX = 01), media (XX = 02) y alta (XX = 03)
sm_mm_XX	Red que aplica las macrooperaciones de *división* y de *fusión* durante su evolución, siendo la *fusión* más frecuente y con tres niveles de intensidad: baja (XX = 01), media (XX = 02) y alta (XX = 03)
sm_sm_XX	Red que aplica las macrooperaciones de *división* y de *fusión* durante su evolución, siendo la *divisón* más frecuente y con tres niveles de intensidad: baja (XX = 01), media (XX = 02) y alta (XX = 03)

Tabla 4.2: *Descripción de las 18 redes dinámicas seleccionados para el caso de uso.*

Operadores	
Función de fitness	Average ODF y Community Score
Codificación	Basada en Locus
Selección	Torneo
Cruce	Cruce en dos puntos
Mutación	Función personalizada que selecciona aleatoriamente varios genes de un individuo y cambia sus valores a un vecino aleatorio
Criterio de parada	Prueba Chi-cuadrado de una sola muestra y número máximo de generaciones

Parámetros comunes	
Tamaño del Torneo	2
Tamaño de la Población	300
Tamaño de la Descendencia	300
Probabilidad de Cruce	100 %
Probabilidad de Mutacion	10 %
Chi-cuadrado de una sola muestra	0.05
Generaciones máximas	300
RIGA y EIGA	
Número de immigrantes (N)	10 %

Tabla 4.3: *Configuración del MOGA del caso de estudio*

Método	Estabilidad	Calidad	Velocidad
RIGA	▼	≈	▲
HIGA	▼	▼	▲
EIGA	▼	▲	▲
HGA	▼	≈	▲
DEIGA	▼	▲	▲

Tabla 4.4: *Resultados de la prueba de rango con signo de Friedman con análisis post-hoc de Holm utilizando el Estándar como método de control. El primer grupo contiene las tres estrategias de inmigrantes, y el segundo el resto de los métodos.* ▲ *indica que un método tiene un mejor rendimiento que el Estándar, por el contrario,* ▼ *indica que un método funciona peor y, por último,* ≈ *indica que el método tiene el mismo rendimiento.*

4.3.4. Aplicación de la metodología

4.3.4.1. Paso uno: Comparación de los métodos bajo estudio con el Estándar

Siguiendo los pasos de la metodología, el primer paso consiste en comparar los métodos bajo estudio con el Estándar, es decir, el MOGA sin inmigrantes. Los resultados obtenidos en esta fase están disponles en la Tabla 4.4.

Analizando dicha tabla podemos ver que los resultados son dispares en cuanto a la *calidad* de las estructuras de comunidad encontradas. Por un lado, de las tres técnicas de inmigrantes bajo estudio, EIGA es la única que mejora el Estándar, mientras que RIGA e HIGA presentan un rendimiento similar y peor, respectivamente. Respecto a los métodos adicionales, DEIGA mejora el Estándar, mientras que HGA exhibe un rendimiento parecido. El hecho de que DEIGA muestre mejor rendimiento que el *Estándar* era de esperar, ya que EIGA también lo hace, y DEIGA no es más que un EIGA con un retraso en la inserción de los inmigrantes. Sin embargo, el hecho de que HGA no mejore los resultados del Estándar revela que la inserción gradual de individuos en una población aporta beneficios frente a la reutilización directa de poblaciones pasadas, al menos cuando se trabaja en el CDP dinámico.

En cuanto a la *velocidad* de los algoritmos, todas las técnicas estudiadas muestran un incremento de *velocidad* frente al *estándar*. Este era un resultado esperado, ya que la principal razón para aplicar un mecanismo de esta índole es mejorar de la velocidad de convergencia de los algoritmos genéticos mediante la reutilización del conocimiento pasado.

Por último, al contrario de lo que ocurría con la dimensión *velocidad*, todos los algoritmos considerados presentan una *estabilidad* peor que el algoritmo *Estándar*. Esto implica que los HVs de los Frentes de Pareto obtenidos por todos los métodos tienen mayores variaciones entre ejecuciones en comparación con los obtenidos por el método de control. Esto, es señal de que la reutilización de individuos de poblaciones pasadas deteriora la capacidad de exploración de un MOGA, haciéndolo más propenso a quedarse atascado en un mínimo/máximo local. Obsérvese que no sólo las estrategias de los inmigrantes sufren una menor estabilidad, sino también HGA y DEIGA.

4.3.4.2. Paso dos: Comparación de los métodos bajo estudio entre sí

El segundo paso indicado en la metodología consiste en realizar un análisis comparativo entre todas las técnicas objeto de estudio mediante la prueba de rango con signo de Friedman y el análisis post-hoc de Shaffer. Los resultados de este segundo análisis pueden verse en la Figura 4.3.

El paso anterior reveló que EIGA y DEIGA son superiores al *Estándar* en cuanto a la calidad de las soluciones encontradas, mientras que RIGA y HGA presentan un rendimiento similar al del método de control y HIGA uno peor. No obstante, los resultados de la Figura 4.3 refinan el experimento previo y revelan que HGA presenta un mejor rendimiento que RIGA a la vez que atribuyen un rendimiento similar a EIGA y DEIGA. Nótese que el nodo de HGA es mayor que el nodo de RIGA en la figura (a) y no hay ninguna arista que una a RIGA con HGA; además, nótese la arista que une EIGA con DEIGA. A la vista de estos resultados, podemos concluir que para este problema en concreto es mejor reutilizar poblaciones anteriores con una fase de mutación que insertar inmigrantes aleatorios durante el proceso de evolución, ya que HGA rinde mejor que RIGA. Sin embargo, no podemos sacar ninguna conclusión sobre si la población inmigrante y la nativa exploran diferentes áreas del espacio de búsqueda, ya que tanto EIGA como DEIGA tienen un rendimiento similar.

Con respecto a la dimensión *Velocidad*, del paso anterior sabemos que todos los métodos son más rápidos que el *Estándar*. Sin embargo, este experimento revela la existencia de dos grupos diferentes de algoritmos, por un lado, están RIGA, HGA y EIGA; nótese que sus nodos forman un cliqué y, por otro lado, están HIGA y DEIGA, cuyos nodos forman una especie de cola. A partir de estos resultados, podemos concluir que la inserción de inmigrantes aleatorios (RIGA) o elitista (EIGA) presenta una velocidad de convergencia similar a la de la reutilización directa de una población anterior (HGA). Sorprendentemente, el método HIGA, que utiliza una combinación de inmigrantes elitistas y aleatorios, presenta una velocidad de convergencia peor que el resto de las técnicas, incluidas las dos que combina.

Por último, respecto a la dimensión *estabilidad*, el paso anterior de la metodología mostró que todos los algoritmos considerados tienen peor *estabilidad* que el *Estándar*. Analizando los resultados obtenidos en este nuevo paso, podemos concluir que todas las técnicas se comportan igual de mal con respecto a esta dimensión; nótese que en el grafo todos los nodos forman un cliqué.

4.3.4.3. Paso tres: Análisis de grano fino para identificar anomalías

Siguiendo el tercer paso de la metodología, en primer lugar, se evalúa el rendimiento de las técnicas de inmigrantes para cada red dinámica en base a la métrica de Calidad, con el fin de identificar qué estrategias tienen un rendimiento bajo. Los resultados de la Tabla 4.5 refuerzan las conclusiones extraídas en los dos pasos anteriores: el uso de inmigrantes elitistas tiene beneficios sobre la reutilización directa de poblaciones pasadas, mientras que el uso de inmigrantes aleatorios no. Cabe destacar que HIGA y RIGA tienen un rendimiento numérico peor que HGA, que a su vez tiene un rendimiento peor que EIGA. Sin embargo, se puede observar que hay varias redes para las que todas las técnicas de inmigrantes bajo estudio obtienen peor *calidad* que el Estándar. En particular, las redes *sm_eq_01*, *sm_eq_02*, y *sm_mm_01* son de especial interés porque todos los métodos fallan excepto DEIGA. Para profundizar en el análisis de los

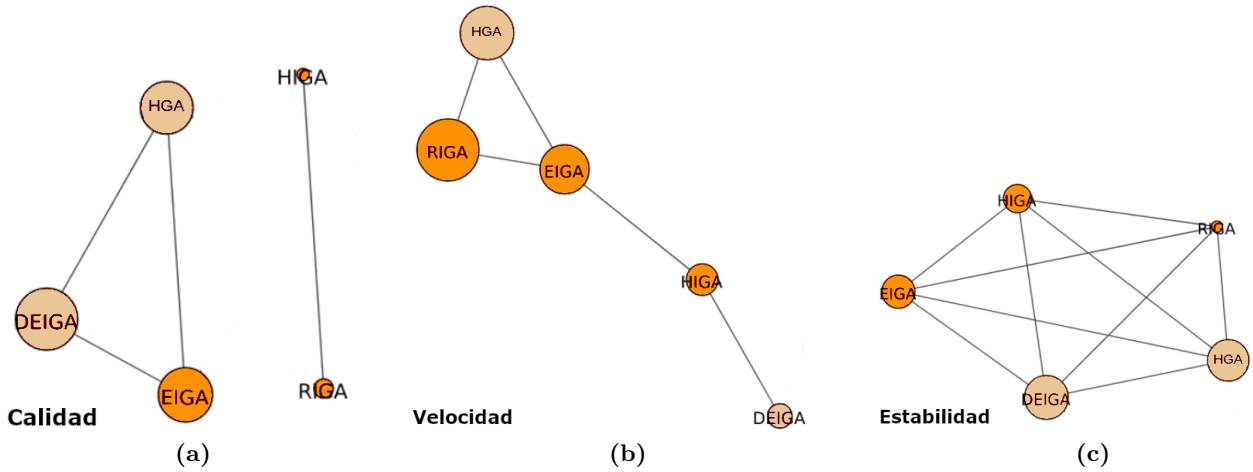


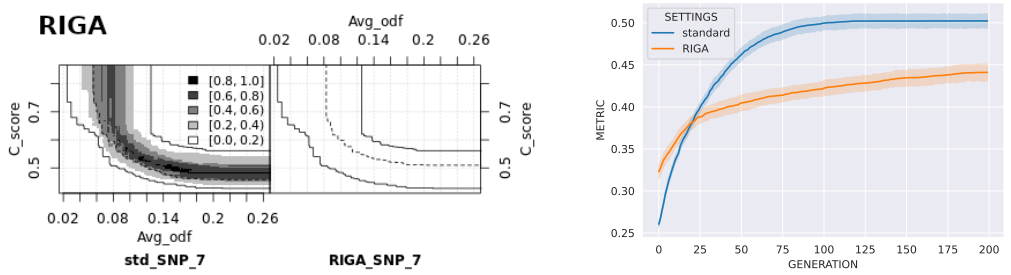
Figura 4.3: *De izquierda a derecha, los resultados de la prueba de rango con signo de Friedman y el análisis post-hoc de Shaffer para las dimensiones de calidad, velocidad y estabilidad. Cada nodo representa un algoritmo diferente. Los nodos de color oscuro corresponden a las técnicas de inmigración en estudio y los claros a los demás métodos. El tamaño de los nodos es inversamente proporcional a su rango en la prueba de Friedman en la dimensión de Calidad y directamente proporcional en las de Velocidad y Estabilidad (es decir, los nodos más grandes tienen un mejor rendimiento). Dos nodos están conectados cuando la diferencia entre dos métodos no es estadísticamente significativa.*

fenómenos anteriores, se pasa a estudiar el comportamiento de cada técnica de inmigrantes en la red *sm_eq_01*. Nótese que, aunque sólo se muestran los resultados de la red *sm_eq_01* las conclusiones obtenidas al analizar las demás redes son las mismas.

Una vez seleccionada una instancia problemática de la batería de pruebas, en nuestro caso *sm_eq_01*, se pasa a analizar la evolución del HV y las attainment surfaces obtenidas en cada instantánea de dicha red, con la intención de identificar aquellas en las que ocurren convergencias prematuras. Los resultados de la Figura 4.4, la cual muestra la instantánea 7 de la red seleccionada, muestran cómo todas las técnicas de inmigrantes, menos DEIGA, sufren de una convergencia prematura que afecta a su capacidad de exploración. Nótese como la curva naranja se estanca en un mínimo local comparada con la azul y la presencia de manchas oscuras en varias zonas de las attainments surfaces. Además, nótese cómo estos efectos son más suaves en EIGA que en RIGA y, cómo a su vez, son más suaves en RIGA que en HIGA. Aunque en este capítulo solo se muestren las gráficas para la instantánea 7 de la red, las mismas conclusiones se pueden extraer de las gráficas de las instantáneas 8 y 9, mientas que en las instantáneas de 1 al 6 no se detecta convergencia prematura alguna en ningún método.

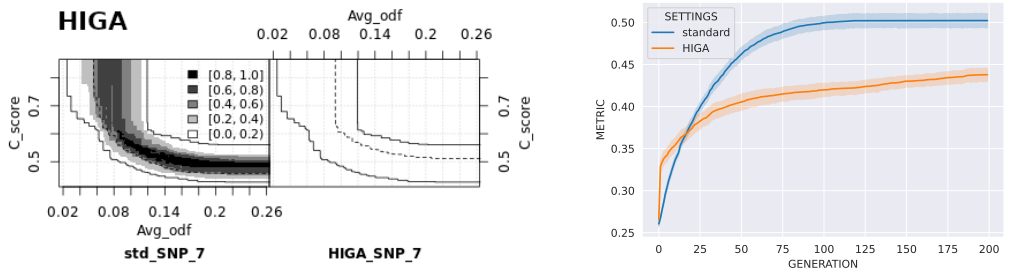
Para profundizar en el estudio de por qué las diferentes técnicas de inmigración no están evitando esta convergencia prematura, se analiza cada método de forma independiente utilizando la técnica de visualización de la mezcla de poblaciones:

Comenzando con la técnica RIGA, como se observa en la Figura 4.5 (pág 74), la mezcla entre nativos (individuos de la instantánea anterior en el caso de RIGA) e inmigrantes (individuos completamente aleatorios en el caso de RIGA) no tienen suficiente calidad y son eliminados casi por completo de la población en la misma generación que se crean. Además, el porcentaje de



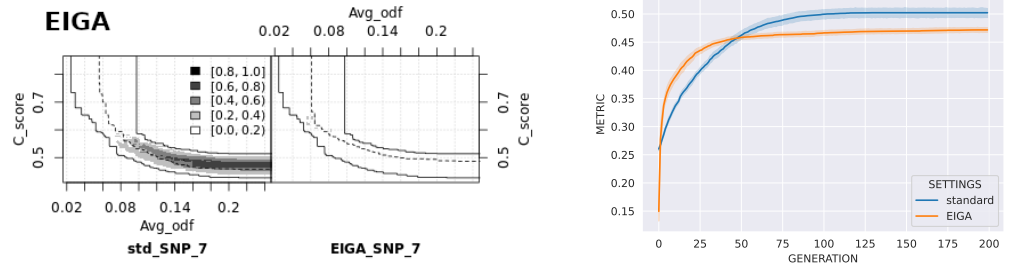
(a)

(b)



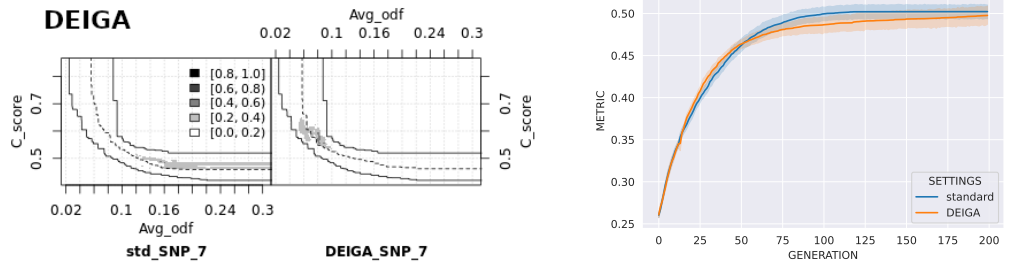
(c)

(d)



(e)

(f)



(g)

(h)

Figura 4.4: *Las figuras de la izquierda (a, c, e, g) muestran las Attaiment Surfaces de los últimos frentes de Pareto encontrado por el Estándar (panel izquierdo) y la estrategia inmigrantes (panel derecho). Las figuras de la derecha (b, d, f, h) muestran la evolución del HV a lo largo de las diferentes generaciones del AGMO para los mismos métodos. El Estándar se muestra en naranja, mientras que la estrategia del inmigrante se muestra en azul.*

Red	HIGA	RIGA	EIGA	HGA	DEIGA
mg_01	▼	▼	≈	≈	≈
mg_02	▼	▼	≈	≈	≈
mg_03	▼	▼	≈	≈	≈
mx_01	≈	≈	▲	▲	▲
mx_02	≈	≈	▲	▲	▲
mx_03	≈	≈	▲	▲	▲
sm_eq_01	▼	▼	▼	▼	≈
sm_eq_02	▼	▼	▼	▼	≈
sm_eq_03	▼	≈	▲	▲	▲
sm_mm_01	▼	≈	▲	≈	≈
sm_mm_02	▼	▼	▼	▼	≈
sm_mm_03	≈	≈	▲	▲	▲
sm_sm_01	▼	▼	≈	≈	▲
sm_sm_02	▼	▼	≈	≈	≈
sm_sm_03	▼	≈	▲	≈	▲
sp_01	▼	≈	▲	▲	▲
sp_02	▼	≈	▲	≈	▲
sp_03	▼	▼	≈	≈	≈

Tabla 4.5: *Resultados de la prueba de rango con signo de Friedman y del análisis post-hoc de Holm sobre el HV de las soluciones y usando el Estándar como método de control.*

inmigrantes insertados en la población no cambia este efecto. Por lo tanto, es difícil que dos individuos inmigrantes se crucen en RIGA y casi imposible que sus descendientes continúen en la siguiente generación. Dado que los inmigrantes son expulsados de la población, sin tener la oportunidad de transmitir sus genes a la siguiente generación, no hacen otra cosa que desperdiciar espacio dentro la población. Por lo tanto, a efectos prácticos, la técnica RIGA en el CDP dinámico es equivalente al método HGA con una población más pequeña, lo que produce una convergencia rápida y una peor calidad en las soluciones.

En cuanto a la técnica HIGA, la Figura 4.6 (pág 75) muestra el número de individuos de cada tipo durante el proceso evolutivo, en la subfigura (a) se muestra los individuos que continúan de una generación a la siguiente y en la (b) los que son expulsados. Si nos centramos en la subfigura (a), podemos ver que el método HIGA inserta un gran número de inmigrantes (individuos de la instantánea anterior) durante las primeras generaciones. De hecho, inserta tantos inmigrantes que la población nativa se extingue después de la quinta generación. Sin embargo, analizando la subfigura (b), se puede observar que después de que la población nativa se haya extinguido, el número de individuos inmigrantes que abandonan la población disminuye y luego se estabiliza. No es descabellado hipotetizar que esta meseta se produce porque el algoritmo intenta mejorar su capacidad de exploración añadiendo inmigrantes aleatorios, que es la misma técnica que utiliza RIGA. Resumiendo, el método HIGA es demasiado agresivo al principio del proceso evolutivo y elimina todos los individuos nativos de la población, comportándose esencialmente como HGA. Después, intenta aumentar la diversidad de la población introduciendo individuos aleatorios, al igual que RIGA, lo que hace que los inmigrantes sólo utilicen espacio y no contribuyan a la exploración (es decir, es como utilizar una población más pequeña). Este comportamiento explica la mala calidad de los resultados de HIGA: utiliza los inmigrantes aleatorios como RIGA, pero,

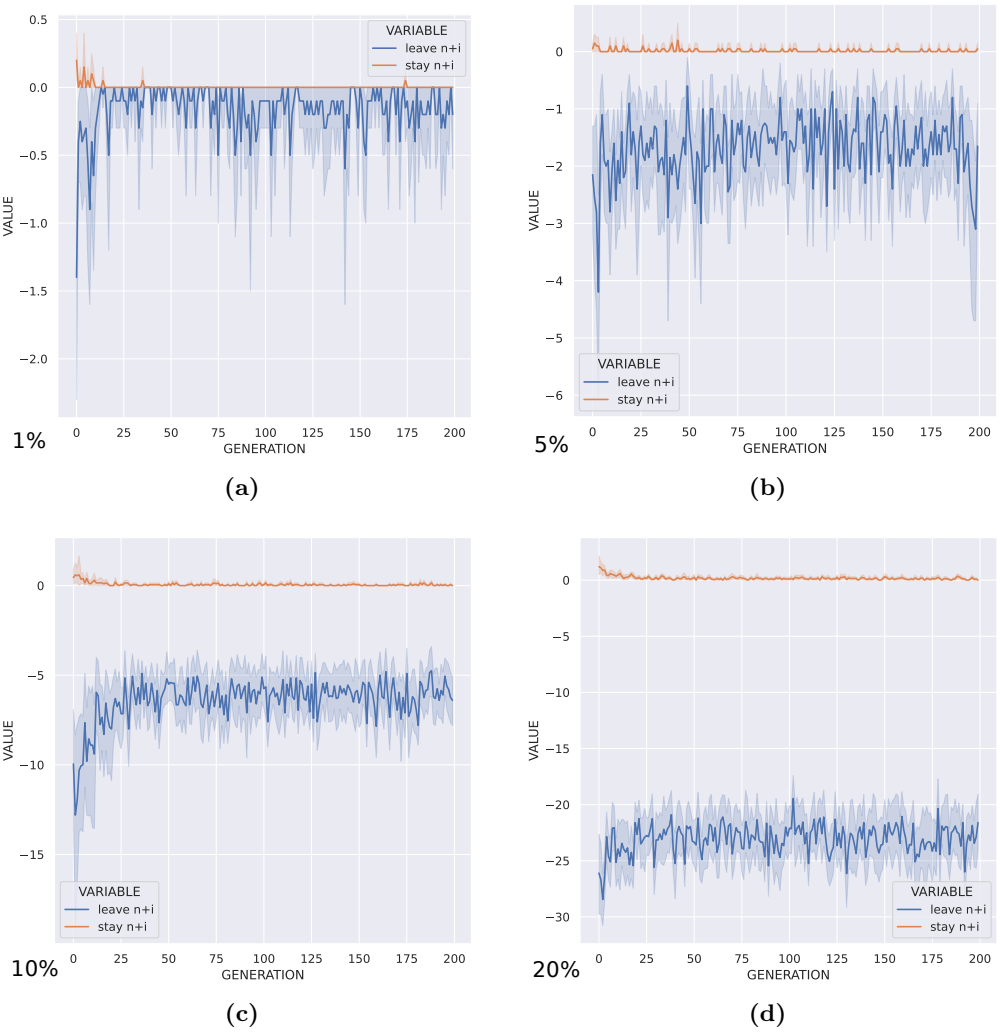
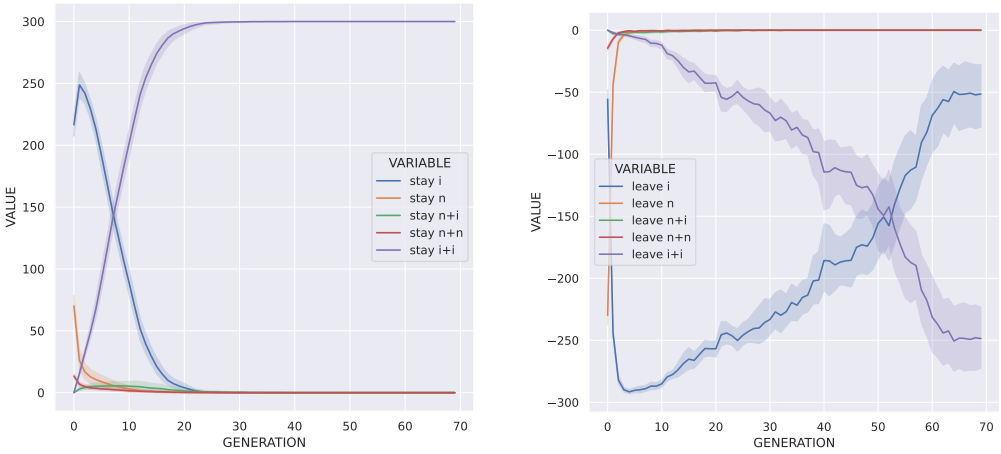


Figura 4.5: *Número de individuos mestizos para el método RIGA que sobreviven (en naranja) y son expulsados (en azul) de la población entre dos generaciones consecutivas. El eje X muestra el número de generaciones y el eje Y el número de individuos de un tipo determinado. El porcentaje en la esquina inferior derecha del gráfico indica el porcentaje de inmigrantes insertados en la población en cada generación.*



(a) **(b)**

Figura 4.6: *Número de individuos de cada tipo para el método HIGA. El eje X muestra el número de generaciones y el eje Y el número de individuos de un tipo determinado. La subfigura (a) muestra el número de individuos que sobreviven de una generación a otra y la subfigura (b) muestra el que son expulsados.*

a diferencia de RIGA, HIGA lo hace gradualmente durante el proceso evolutivo, de ahí que la velocidad de convergencia sea menor.

Con respecto al EIGA, la Figura 4.7 (pág 76) muestra el número de individuos de diferentes tipos para una generación concreta. A diferencia de los otros métodos, en EIGA sí que hay una mezcla adecuada entre individuos nativos e inmigrantes; nótese la presencia de individuos mestizos en su gráfica de mezcla de poblaciones (la línea verde n+i de la figura). Como era de esperar, hay una fase de transición en la que las dos poblaciones (inmigrante y nativa) se cruzan, produciendo descendencia válida. Tras ese periodo, ambas poblaciones siguen mezclándose, pero dan lugar a soluciones de menor calidad que son expulsadas de la población. La duración de este periodo de mezcla varía en función del porcentaje de inmigrantes insertados en la población; cuanto mayor sea el porcentaje de inmigrantes, más corto será el periodo. Además, para porcentajes tan altos como el 5 %, la población inmigrante (o sus descendientes) terminan por conquistar la población nativa (subfiguras (b), (c) y (d)). Sin embargo, cuando el porcentaje es del 1 % (subfigura (a)), se observa que ninguna población domina totalmente a la otra y el proceso evolutivo alcanza un equilibrio en el que la población inmigrante ocupa una parte de la población, mientras que la nativa ocupa el resto. Este hecho sugiere que la población inmigrante y la nativa podrían explorar diferentes regiones del espacio de búsqueda.

Por último, se analiza DEIGA, el cual se incluyó por razones comparativas. Como ya se ha dicho, este método funciona como EIGA, pero la inserción de los inmigrantes se retrasa un cierto número de generaciones. A partir de los resultados mostrados en la subfigura (a) de la Figura 4.7 (pág 76), se puede hipotetizar que, si se deja evolucionar a la población nativa durante un cierto número de generaciones, para que pueda adaptarse mejor al entorno, la mezcla de individuos podría producir mejores resultados. El método DEIGA permite comprobar esta hipótesis. La Figura 4.8 muestra el número de individuos de diferentes tipos durante el proceso evolutivo de DEIGA. Como se puede ver, incluso esperar un pequeño número de generaciones (por ejemplo, 10) impide que la población inmigrante conquiste la población nativa (nótese la

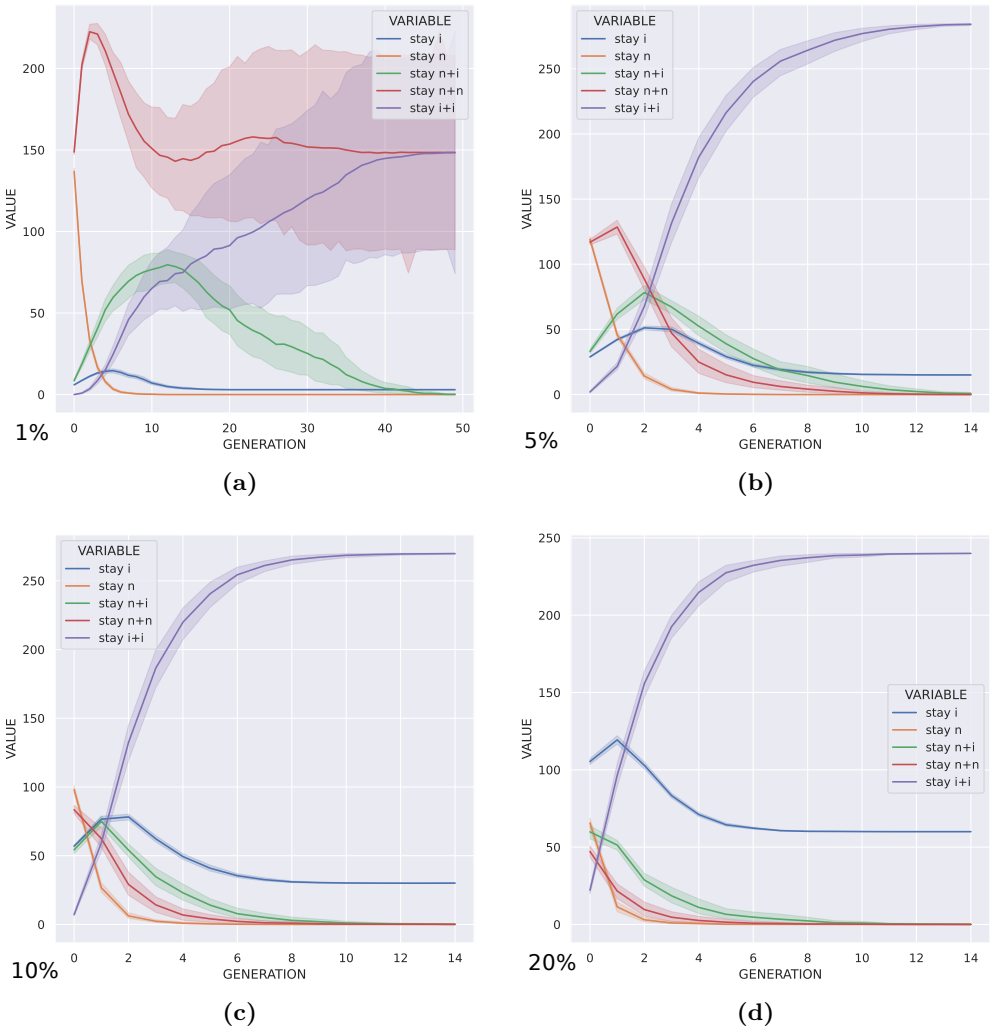


Figura 4.7: *Número de individuos de cada tipo para el método EIGA. El eje X muestra el número de generaciones que dura el proceso evolutivo y el eje Y el número de individuos de un tipo concreto. El porcentaje en la esquina superior derecha del gráfico indica el porcentaje de inmigrantes insertados en la población en cada generación.*

diferencia con la Figura 4.7 (c)). Además, cuanto mayor sea el número de generaciones que se espera, menor será la diferencia de tamaño entre las poblaciones nativa e inmigrante cuando se alcance el equilibrio. Sin embargo, si el valor es demasiado grande, como en la subfigura (e), se produce el efecto contrario ya que los inmigrantes de la población no tienen suficiente calidad para ocupar su propio espacio en ella. Aunque cabe resaltar, que siempre ocurre una mezcla entre las poblaciones cuando se empiezan a introducir inmigrantes (nótese que la línea verde n+i siempre está presente en todos los gráficos). En resumen, dejar que la población nativa evolucione algunas generaciones antes de insertar inmigrantes produce los mejores resultados, ya que evita que los inmigrantes se apoderen de toda la población y permite una buena mezcla entre las poblaciones. Sin embargo, esperar demasiadas generaciones produce el efecto contrario e impide que los inmigrantes entren en la población. No obstante, incluso en estos casos, siempre aparece algún tipo de mezcla entre las poblaciones.

4.3.5. Conclusiones del caso de uso

Tras aplicar la metodología podemos concluir que que tanto RIGA como HIGA tienen un bajo rendimiento numérico. El análisis de grano fino realizado ha revelado que la inserción de individuos completamente aleatorios en una población ya establecida produce una descendencia de tan poca calidad, en comparación con el resto, que se descarta inmediatamente de la población. En la práctica, estos métodos reducen el tamaño de la población, ya que las soluciones inmigrantes se descartan sin contribuir a la capacidad de exploración del algoritmo. Sin embargo, el método EIGA muestra mejores resultados, ya que es el único en el que se consigue una mezcla adecuada entre la población nativa y la inmigrante. No obstante, el porcentaje de inmigrantes insertados en la población afecta a la calidad de los resultados numéricos. Si se insertan demasiados inmigrantes, la presión de selección de estos es demasiado alta y conquistan toda la población, haciendo que la población nativa se extinga. Sorprendentemente, cuando el porcentaje de inmigrantes insertados no es demasiado alto, se alcanza un equilibrio y ambas poblaciones siguen evolucionando de forma independiente.

Para estudiar más a fondo este fenómeno se evaluó una variante del algoritmo EIGA que incluye un desfase en la inserción de inmigrantes (DEIGA). Los experimentos han demostrado que esperar tan sólo diez generaciones antes de insertar inmigrantes en la población nativa produce un equilibrio entre ambas poblaciones que siguen evolucionando de forma independiente, incluso para porcentajes elevados de inserción de inmigrantes como el 10 % de la población. Además, se ha comprobado que cuanto más se espera antes de insertar los inmigrantes, menor es la diferencia de tamaño entre ambas poblaciones durante el equilibrio. Sin embargo, si se esperan demasiadas generaciones, se produce el efecto contrario y los individuos inmigrantes no tienen suficiente calidad para entrar en la población. No obstante, en todos los experimentos independientemente del tiempo de espera siempre hay una mezcla entre las poblaciones.

La presencia de este equilibrio, en el que ambas poblaciones evolucionan en paralelo, sugiere que, para esta configuración particular (MOGAs para la detección de comunidades dinámicas), las soluciones inmigrantes y las nativas tienen diferentes capacidades de exploración y, para lograr los mejores resultados, la población nativa necesita cierto tiempo para madurar. Por esta razón, es crucial que la presión de selección del algoritmo no expulse las soluciones nativas del proceso evolutivo demasiado rápido. Teniendo en cuenta los puntos anteriores, podemos plantear la hipótesis de que hay pruebas que respaldan que un algoritmo genético multipoblacional podría ser beneficioso para este problema en concreto.

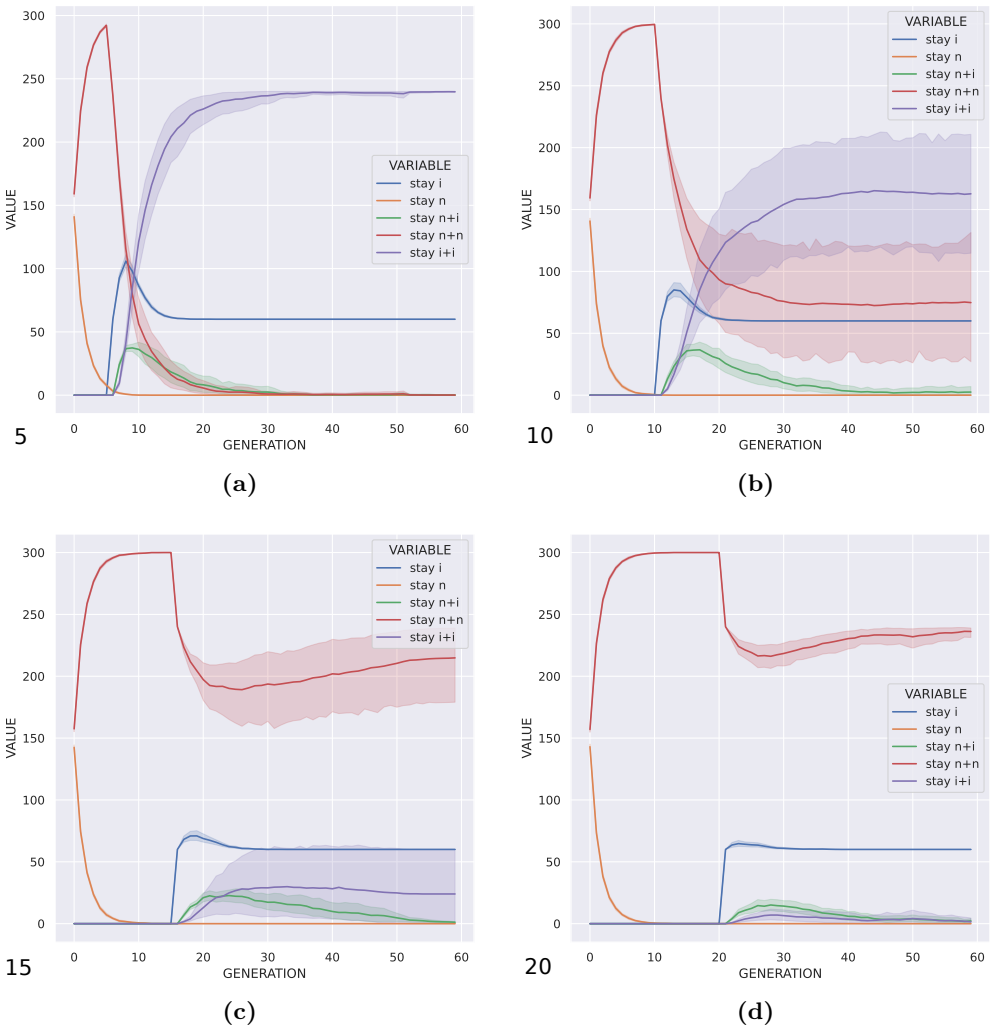


Figura 4.8: *Número de individuos de cada tipo para el método DEIGA. El eje X muestra el número de generaciones durante el proceso evolutivo, mientras que el eje Y muestra el número de individuos de un tipo concreto. El número en la esquina superior derecha del gráfico indica el número de generaciones esperadas hasta que los inmigrantes se insertan en la población. En todos los gráficos se inserta un 10 % de inmigrantes en cada generación.*

4.4. Eficacia y limitaciones de la metodología

En este capítulo se ha presentado una metodología nueva para evaluar estrategias de inmigrantes en MOGAs en base a tres dimensiones diferentes: Calidad, Estabilidad y Velocidad. El objetivo de esta metodología es doble. Por un lado, permite evaluar si una determinada estrategia de inmigrantes está mejorando o perjudicando alguna de las dimensiones mencionadas, y a la vez, también permite obtener información sobre lo que podría estar ocurriendo en el caso de que alguna de las dimensiones se vea afectada negativamente. Para demostrar la eficacia de la nueva metodología, se ha llevado a cabo un caso de uso en el que se aplica la metodología al CDP dinámico. En nuestro caso de estudio, analizamos los resultados de las tres estrategias de inmigrantes más comunes: RIGA, EIGA e HIGA, además de otros dos métodos, HGA y DEIGA, usados para fijar una referencia de comparación.

El análisis experimental ha demostrado que la inserción de inmigrantes aleatorios, en una población ya establecida, produce progenie de baja calidad que no sobrevive. En la práctica, para este caso de uso concreto, utilizar inmigrantes aleatorios equivale a reducir el tamaño de la población y es mejor reutilizar directamente una población previa. En consecuencia, RIGA e HIGA tienen un bajo rendimiento cuantitativo, incluso en comparación con HGA. Por el contrario, el estudio muestra que el uso de inmigrantes elitistas consigue resultados prometedores y produce beneficios sobre la reutilización directa de poblaciones pasadas. De ahí el mejor rendimiento de EIGA sobre todos los demás métodos, incluido el método de referencia HGA. Además, el análisis experimental ha mostrado que en EIGA se alcanza un equilibrio entre las poblaciones inmigrantes y nativas, lo que sugiere que un algoritmo genético multipoblacional sería beneficioso para esta estrategia y problema concreto. Por lo tanto, tras el caso de uso, podemos concluir que la metodología propuesta ha permitido sistematizar el análisis de las diferentes técnicas de inmigrantes y que la técnica de visualización de la mezcla de poblaciones proporciona información valiosa sobre su funcionamiento interno. Finalmente, teniendo en cuenta lo anterior, podemos declarar que nuestra metodología podría facilitar la aplicación exitosa de las técnicas de inmigrantes a los MOGAs.

Sin embargo, hay cuatro áreas que merecen un mayor estudio que el recibido en este capítulo:

- Diseñar un MOGA multiplobacional para el CDP dinámico dónde dos poblaciones, una con los individuos nativos y otra con los inmigrantes, exploraran el área de búsqueda de forma independiente intercambiando soluciones de manera ocasional.
- Estudiar el CDP dinámico desde la perspectiva de una Evolutionary Multi-Task Optimisation (Optimización evolutiva multitarea) (EMTO). En EMTO, en lugar de resolver cada tarea de una en una, todas las tareas se resuelven al mismo tiempo, utilizando el conocimiento adquirido en una tarea para ayudar a las demás. En particular, es de especial interés el algoritmo MFEA [40] en el que todas las tareas (instantáneas, en nuestro caso de estudio) son optimizadas al mismo tiempo por una única población.
- Extender el caso de uso presentado en este capítulo a problemas dinámicos más complejos, como los problemas continuos, con el fin de evaluar más exhaustivamente las capacidades y limitaciones de dicha metodología, dado que el CDP es un problema discreto.
- Adaptar también la metodología a problemas dinámicos mono-objetivo, dado que la metodología propuesta sólo permite sistematizar el análisis en el caso de problemas dinámicos multi-objetivo.

Capítulo 5

Conclusiones y trabajo futuro

En este capítulo se resumen las conclusiones adquiridas durante la realización de la tesis, así como las posibles líneas de investigación futura que continúen el trabajo presentado en este tomo. Se comienza haciendo un resumen de los principales hitos alcanzados en esta tesis para, seguidamente, pasar a responder las tres preguntas de investigación planteadas en el capítulo uno de este documento (Sección 5.1). Finalmente, se presentan varias líneas de trabajo futuro que podrían suponer la continuación del trabajo de investigación desarrollado en esta tesis doctoral (Sección 5.2).

5.1. Conclusiones

La presente tesis se ha centrado en la aplicación de MOGAs para la detección de comunidades dinámicas; en concreto, en estudiar aquellas técnicas que permiten reducir el tiempo de ejecución en aras de hacer estos algoritmos aplicables a problemas reales y de mayor complejidad. Se identificaron principalmente dos escollos para aplicar un MOGA al CDP dinámico: (i) la necesidad de adaptar los individuos a los cambios en la topología de la red en aras de evitar la aparición de soluciones inválidas y (ii) la necesidad de adaptar las poblaciones a cambios en la red para evitar que pierdan capacidades.

Respecto a la problemática de la aparición de soluciones inválidas, en el Capítulo 3 se han propuesto tres operadores capaces de reparar un individuo, es decir, transformar una solución inválida en una solución válida de nuevo. A la hora de realizar el proceso de reparación los mecanismos propuestos son capaces de aprovechar el conocimiento pasado de tal manera que apenas se pierde información durante el proceso de reparación. Para probar la validez de dichos mecanismos, se ha llevado a cabo un caso de uso dónde se probaron los operadores de reparación en un MOGA para la detección de comunidades dinámicas. Dicho caso de estudio analizó el rendimiento del MOGA usando una batería de pruebas compuesta por 66 redes sintéticas y dos reales.

En relación con el segundo gran problema detectado, en el estado del arte se pueden encontrar varias técnicas que permiten adaptar una población a un cambio del entorno, de tal manera que no pierda capacidades y, entre ellas, las técnicas de inmigrantes son de las más populares. Aunque implementar dichas técnicas en un algoritmo genético es un proceso sencillo; su posterior

ajuste no lo es tanto, ya que se tienen que configurar una serie de parámetros que pueden afectar de manera crítica al rendimiento del algoritmo. En aras de hacer este proceso más sencillo, en el Capítulo 4 se propone una metodología que permite evaluar estrategias de inmigrantes en MOGAs en base a tres dimensiones diferentes: Calidad, Estabilidad y Velocidad. La metodología se diseñó con un propósito doble: por un lado, evaluar si una determinada estrategia de inmigrantes está mejorando o perjudicando alguna de las dimensiones mencionadas y, por otro, dar información sobre lo que podría estar ocurriendo en el caso de que alguna de las dimensiones se vea afectada negativamente. Finalmente, para probar la validez de esta metodología, se llevó a cabo un caso de uso en el que se evaluaron tres de las estrategias de inmigrantes más populares usando 18 redes dinámicas sintéticas.

Teniendo en cuenta todos los análisis experimentales realizados en los capítulos 3 y 4, es posible dar respuesta a las principales preguntas de investigación presentadas en la sección de Introducción que se pasan a responder a continuación:

RQ1: *¿Cuál es el método más eficiente para adaptar los individuos de un GA diseñado para la detección de comunidades a los cambios del entorno?*

Dependiendo del problema, es posible que un cambio en el entorno provoque que un individuo que codifica una solución completamenta válida degenere y pase a codificar una solución imposible o que inclumpla alguna restricción del problema; cuando esto ocurre decimos que un individuo es inválido. Al proceso de transformar un individuo inválido en uno válido de nuevo se le conoce como proceso de reparación. El estado del arte no contenía trabajos que tratasen la aparición de soluciones inválidas en el CDP dinámico. Por esa razón, durante la realización de esta tesis, se han desarrollado y probado tres técnicas que permiten reparar una solución que se haya vuelto inválida a raíz de un cambio en una red. Se propusieron tres métodos: uno basado en *Label Propagation*, otro basado en *Random Walks* y un último *Híbrido* de los dos anteriores. La experimentación demostró que la mejor forma de reparar individuos era usar el método basado en *Random Walks*, ya que este mostraba un rendimiento superior al método basado en *Label Propagation* y un rendimiento equivalente al obtenido por el *Híbrido*, pero siendo más ligero en cuanto a consumo de recursos se refiere. Además, el análisis del caso de uso arrojó resultados muy positivos al reparar poblaciones usando el operador basado en *Random Walks*; de media se puede esperar una reducción del tiempo de ejecución de un algoritmo entorno al 50 %, sin perder apenas calidad en las soluciones encontradas. Eso sí, también se detectó un incremento en la dispersión de la calidad de las soluciones encontradas en ejecuciones consecutivas, lo que implica una pérdida de estabilidad, ya que es posible que los algoritmos que inlcuyen estos mecanismo de reparación de individuos retornen soluciones de peor calidad con mayor frecuencia.

RQ2: *Entre las estrategias presentes en el estado del arte para adaptar la población de un GA a cambios en el entorno ¿Cuál de ellas funciona mejor para la versión dinámica del CDP y qué ventajas e inconvenientes presenta?*

Respecto a la mejor estrategia para adaptar una población a un cambio, en el estado del arte hay propuestos muchos mecanismos para llevar a cabo dicha tarea, entre los que cabe destacar la hipermutación, las estrategias de memoria o las estrategias de inmigrantes. Esta última es la más popular de las todas. Nótese que aunque todas las técnicas han sido

estudiadas en otros dominios de aplicación diferentes al CDP, estas son lo suficientemente genéricas para aplicarse en él sin demasiado problema. Durante la realización de esta tesis se probaron tres de las estrategias de inmigrantes más populares: RIGA, EIGA e HIGA. La experimentación muestra que la estrategia de inmigrantes elitistas (EIGA) supera con creces a las otras dos, siendo esta igual de veloz que la técnica de hipermutación (HGA, la más simple de todas ya que consiste en reutilizar la población previa tras someterla a un proceso de mutación, pero retornando soluciones de mayor calidad. Además, análisis de grano fino posteriores mostraron que los problemas con las técnicas de RIGA e HIGA radican en el uso de inmigrantes aleatorios, ya que estos al introducirse en una población completamente formada producen descendientes de tan poca calidad que son expulsados inmediatamente de la población, lo que a efectos prácticos equivale a utilizar un tamaño de población menor. Por último, estos análisis de grano fino también mostraron que en EIGA la población inmigrante y nativa es capaz de explorar espacios diferentes del área de búsqueda, ya que sí se dejaba madurar a la población nativa unas cuantas generaciones antes de empezar a meter inmigrantes, ambas poblaciones llegaban a un equilibrio. Este fenómeno parece sugerir que una estrategia multipoblación sería beneficiosa a la hora de resolver este problema.

RQ3: *Del conocimiento obtenido durante la realización de esta tesis ¿Cuál es generalizable a otros dominios? es decir ¿Cómo puede elegirse la mejor estrategia para adaptar las poblaciones de un GA en otros problemas dinámicos ajenos al CDP dinámico?*

Aunque el objetivo de esta tesis está centrado en el estudio del CDP dinámico, algunas de las lecciones aprendidas durante la realización de la misma son generalizables a otros problemas dinámicos. En concreto, al buscar la mejor estrategia de inmigrantes a usar para adaptar una población a un cambio, se vio que implementar estas estrategias es bastante sencillo y directo pero que configurarlas era una tarea compleja. No había definida una manera estándar de comparar técnicas de inmigrantes entre sí, ni estaba claro qué se podía esperar de ellas ni qué parámetros era necesario tocar cuando alguna técnica no tenía un rendimiento satisfactorio. Bastante del conocimiento adquirido durante el proceso de ajuste realizado para el CDP dinámico es extrapolable a otros dominios dinámicos y las lecciones aprendidas en estos problemas bien pueden ser usadas por otros investigadores que se enfrenten a la misma tarea, pero en dominios diferentes. Por esa razón, se decidió sistematizar la experimentación llevada a cabo durante el proceso de configuración de la mejor técnica de inmigrantes para el CDP dinámico y sintetizarla en una metodología fácil de implementar, para que de esa forma cualquier investigador que se encuentre con la misma disyuntiva en otros problemas pueda aplicarla. En especial, se descubrió que las dinámicas entre las poblaciones nativa e inmigrante durante el proceso evolutivo era de vital importancia y que no se podía dar por supuesto que simplemente por el hecho de introducir inmigrantes en una población implicase una sinergia entre ambas poblaciones. Para conocer dichas dinámicas se ha diseñado una técnica de visualización que permite analizar las dinámicas de ambas poblaciones, la nativa y la inmigrante, durante el proceso evolutivo de una manera gráfica y sencilla, con la intención de que ayude al investigador a tomar una decisión informada sobre qué parámetros modificar.

5.2. Trabajo Futuro

Por último, se describen brevemente algunas de las líneas de investigacón que podrían ampliar el trabajo presentado en esta tesis. Dichas líneas se pueden dividir en dos grandes categorías: a) aquellas que mejoran los MOGAs para la detección de comunidades dinámicas y b) aquellas que amplían la metodología que permite comparar y configurar estrategias de inmigrantes.

- Respecto a las líneas de trabajo futuro centradas en el CDP dinámico:
  - Como se mostró a lo largo de la diversa experimentación realizada, hay ciertas evidencias que sugieren que un método multi-poblacional sería beneficioso a la hora de adaptar las poblaciones de los algoritmos genéticos a los cambios de una red. Dicho algoritmo debería generar dos poblaciones, una nativa con individuos aleatorios y una inmigrante con los individuos de la instantánea anterior.
  - Siguiendo la lógica presentada en el punto anterior, sería interesante estudiar el CDP dinámico desde la perspectiva de una EMTO. En EMTO, en lugar de resolver cada tarea de una en una, todas las tareas se resuelven al mismo tiempo utilizando el conocimiento adquirido en una tarea para ayudar a las demás. En particular, es de especial interés el algoritmo MFEA [40] en el que todas las tareas (instantáneas en el caso del CDP dinámico) son optimizadas al mismo tiempo por una única población.
  - La experimentación realizada durante el desarrollo de esta tesis mostró que reutilizar poblaciones pasadas conlleva un decremento sistemático de la estabilidad de los algoritmos, independientemente de la técnica usada. Sin embargo, en esta tesis no se ha medido cuánto afecta está pérdida de estabilidad ni se han estudiado posibles técnicas para solventar este problema.
- Respecto a las líneas de trabajo futuro centradas en la metodología para la configuración de estrategias de inmigrantes:
  - Durante la realización de esta tesis se ha presentado un caso de uso donde se aplicaba la metodología propuesta al problema del CDP dinámico, el cual es un problema discreto. Sería interesante extender dicho caso de uso y aplicarlo a dominios más complejos, como pueden ser los dominios reales, para poder validar cómo de generalizable es la metodología propuesta.
  - Toda la experimentación y el diseño llevado a cabo en este trabajo se ha centrado en MOGAs. Desde mi punto de vista, tiene sentido centrarse en algoritmos multi-objetivo debido a la capacidad de estos algoritmos para trabajar con múltiples objetivos que afectan al CDP. Sin embargo, en otros dominios (donde únicamente se trabaje sobre un único objetivo, o la combinación lineal de objetivos relacionados) los algoritmos mono-objetivo tendrán más relevancia que los multi-objetivo, por lo que sería interesante adaptar la metodología propuesta para trabajar con problemas mono-objetivo.

Apéndices

Publicación R1 (versión sin el formato de la revista)

THE FOUR DIMENSIONS OF SOCIAL NETWORK ANALYSIS: AN OVERVIEW OF RESEARCH METHODS, APPLICATIONS, AND SOFTWARE TOOLS*

David Camacho
david.camacho@upm.es
Departamento de Sistemas Informáticos
Universidad Politécnica de Madrid, Spain

Ángel Panizo-LLedot
angel.panizo@urjc.es
Computer Science Department
Universidad Rey Juan Carlos, Spain

Gema Bello-Orgaz
gema.borgaz@upm.es
Departamento de Sistemas Informáticos
Universidad Politécnica de Madrid, Spain

Antonio Gonzalez-Pardo
antonio.gpardo@urjc.es
Computer Science Department
Universidad Rey Juan Carlos, Spain

Erik Cambria
cambria@ntu.edu.sg
School of Computer Science and Engineering
Nanyang Technological University, Singapore

January 17, 2022

ABSTRACT

Social network based applications have experienced exponential growth in recent years. One of the reasons for this rise is that this application domain offers a particularly fertile place to test and develop the most advanced computational techniques to extract valuable information from the Web. The main contribution of this work is three-fold: (1) we provide an up-to-date literature review of the state of the art on social network analysis (SNA); (2) we propose a set of new metrics based on four essential features (or *dimensions*) in SNA; (3) finally, we provide a quantitative analysis of a set of popular SNA tools and frameworks. We have also performed a scientometric study to detect the most active research areas and application domains in this area. This work proposes the definition of four different dimensions, namely *Pattern & Knowledge discovery*, *Information Fusion & Integration*, *Scalability*, and *Visualization*, which are used to define a set of new metrics (termed *degrees*) in order to evaluate the different software tools and frameworks of SNA (a set of 20 SNA-software tools are analyzed and ranked following previous metrics). These dimensions, together with the defined degrees, allow evaluating and measure the maturity of social network technologies, looking for both a quantitative assessment of them, as to shed light to the challenges and future trends in this active area.

***K*eywords** Social Network Analysis · Social Media Mining · Social Data Visualization · Data Science · Big Data

1 Introduction

Currently, online social networks (OSNs) are seen as an essential element for interpersonal relationships in a large part of the world. OSNs allow the elimination of physical and cultural barriers through the globalization of the technology. For this reason, OSNs have billions of active users around the world. OSNs can be defined as a social structure made up of people, or entities, connected by some type of relationship or common interest (professional relationship, friendship, kinship, etc.). From [1], an OSN can be defined as: "a service that allow individuals to (1) define a public (or semi-public) profile within an application or specific domain (friendship, professional, common interests, etc.), (2) manage a list of other users with whom the individual (or entity) will share a connection, and (3) view and traverse their list of connections and those made by others within the social site". Although the origins correspond mainly to

different areas from Social Sciences, the term is attributed to the British anthropologists Alfred Radcliffe-Brown [2, 3] and John Barnes [4].

OSNs were created in the late 90s, when Randy Conrads founded *Classmates* in 1995 and Andrew Weinreich launched, in 1997, *SixDegrees.com*. The first mathematical and statistical tool used for studying networks was developed in sociology [5]. However, from the last decades, and thanks to the increasing capacity of computing systems in both computational power and information storage, new disciplines have contributed to the area of OSNs, from Sciences and Engineering (such as Computer Science, Mathematics, Physics, or Biology) to Social Sciences (such as Sociology, Anthropology, Psychology, Linguistics, or Advertising).

The fast growth of OSN sites has led to an enormous interest in the analysis of this type of networks (the interconnections that originate, their structure, the evolution of the network, the information flow and dissemination, or the patterns that can be extracted from them, among many others). The easy access to this type of information, the availability of vast amounts of data, the simple and straightforward codification in form of graph-based representation, as well as the direct application of any practical results drawn from them, has made OSNs one of the hot research areas in several disciplines as Data Mining [6], Big Data [7], Machine Learning [8], Information Visualization [9], or Complex systems [10], among many others. This growth has been exponential in terms of the number of people connected to the network, (currently, there are around 3.484 billion of active social media users, up 9% year-on-year, connected to different OSNs [11]), as well as, in the number of different social-based applications, tools and frameworks available to analyze this social data. However, the current growth of social media has caused serious problems for traditional data analysis algorithms and methods (such as data mining, statistics or machine learning) [12, 13]. The just mentioned areas have to face the challenge of designing and implementing new methods able to work efficiently with the huge amount of data generated in the OSN.

This exciting area generates thousands of papers per year, hundred of different algorithms, tools, and frameworks, to tackle the challenges and open issues related to OSNs. Therefore, when anyone tries to develop a comprehensive analysis of the state of the art in such a complex and multidisciplinary area, it is necessary to establish a formal method to allow the analysis of the immense amount of information available. To do that, it has been carried out a scientometric analysis (Section 2), which has led us to organize the research process of this paper. This process can be summarized as follows:

1. We have performed a scientometric study over the papers published in the last 5 years to extract both, the most active research areas and the most relevant application domains in SNA. The research areas selected have been: graph theory and network analytics, community detection algorithms, information diffusion models, user profiling, topic extraction, and finally sentiment analysis area. The selected application domains have been: Healthcare, Marketing, Tourism & Hospitality, and Cybersecurity. This analysis has allowed us also to find a set of emerging areas, and we have selected and studied the areas of Politics, detection of fake news, and Multimedia in OSNs. These emerging areas have been selected due to their current research activity level, and their high potential impact in the next years.
2. Once the SNA application domains and fundamental research areas have been selected, we have analyzed in detail the most relevant research works published in the last years in the corresponding research fields and application domains.
3. Due to the complexity of the field, and the increasing amount of available technologies for OSN, we have selected, studied, and analyzed an extensive set of SNA software tools and frameworks.
4. We have defined a set of metrics, which allow any researcher to assess any SNA framework, tool or algorithm. These metrics are based on some key aspects, or features, that are relevant for any algorithm belonging to SNA research field (such as knowledge discovery, information fusion or visualization among others).
5. Finally, we have assessed some relevant frameworks and tools available on the Internet to perform SNA tasks by using the metrics previously mentioned. This assessment allows us to rate the degree of maturity of SNA technologies, so any engineer or researcher can better understand the strengths and weaknesses of the different tools and frameworks currently available.

The main contributions of this paper can be summarized as follows:

1. A detailed review of the current state of the art of a set of highly relevant research works, grouped in different categories depending on the fundamental research area addressed, and the application domain. The different research areas, and application domains, have been selected according to a scientometric analysis, which takes into account the keyword (#hashtag) "*#social network analysis (SNA)*", used in a set of relevant papers published in the last 5 years, along with some seminal papers from these areas.

2. The definition of four new "SNA-*Dimensions*". These dimensions are inspired by the popular V-models [14] used in the Big Data area, and its goal is to measure the capacity of the different frameworks and tools to perform SNA tasks. These dimensions will be used to define a set of metrics (that we named *degrees*), that will allow any researcher to identify the technology maturity or technology readiness level, and the main challenges and trends, under the area of SNA. The dimensions defined will be directly related to *Pattern & Knowledge discovery, Information Fusion & Integration, Scalability*, and *Visualization* research topics. These dimensions try to answer the following Research Questions:

   RQ1) **What can I discover?** (*Pattern & Knowledge discovery*) [12, 13]: This is the most classical and studied characteristic in OSNs. From the past century different methods and models from Social Sciences, and more recently from Science and Engineering, have been used to discover new knowledge and patterns (in form of network structure and topology, statistical correlations, information flow and diffusion, etc.) in networks. We will define this feature as a dimension related to the capacity of algorithms, methods and techniques to gather knowledge (usually complex and non-trivial patterns) from OSNs.

   RQ2) **What is the limit?** (*Scalability*): [15, 16]: Due to the fast growth of OSNs, this is a critical dimension based on the capacity of algorithms, methods and frameworks to work with large amounts of data in an adequate time (when the timing would be necessary for a particular application, as could happen in Marketing or Recommender Systems in OSNs). As occur with *Volume* dimension in Big Data [17], this dimension joins both computational time and volume of data to obtain a qualitative measure related to the capacity to manage large networks.

   RQ3) **What kind of data can I integrate?** (*Information Fusion & Integration*) [18, 19]: This dimension will be related to the capacity of fusing different kinds of sources (text, video, images, audio). The currently available OSNs provide data in different formats, which allow defining different types of networks (as Multilayer SNA), which can be later integrated to generate new knowledge.

   RQ4) **What can I show?** (*Visualization*) [20, 21]: Visualization in OSN is one of the most powerful tools used in this area to analyze the knowledge and patterns hidden in these networks. Due to the huge amount of available data, and the complex information (and patterns), which can be stored in a network, an adequate visualization of this information is always a challenging task for any OSN analyst. For this reason, the capability to visualize, filter and represent adequately the information stored in a network will be one of the dimensions selected.

3. The definition of a new *global Capability metric*, named $\mathfrak{C}_{SNA}$, based on the just mentioned metrics that can be used to rank the capabilities of the SNA technology, framework or tool analyzed.

4. The assessment of 20 of the most relevant frameworks and tools to perform SNA tasks using these different dimensions and the global Capability metric defined. This analysis not only shows what are the main strengths and weaknesses of the different frameworks, but it also provides a useful guide for those beginners or senior researchers in the area of SNA.

To help readers through the contents of this paper, Fig. 1 shows the overall organization of this work. Using this figure, readers can easily understand the main contents of each section and go directly to those contents that can be more relevant for their future work.

This paper is structured as follows: Section 2 contains the scientometric analysis performed to highlight the different research areas with high impact in SNA research field. Section 3 provides an analysis of the different concepts and research works that belongs to the SNA research field from the Computer Science point-of-view. In this sense, there are two approaches based on what kind of information is used to perform the analysis: the structure of the network, or the content of the messages published. The different application domains in SNA, as well as the emerging areas, are analyzed in Section 4. Section 5 provides a discussion and some conclusions from the state of the art analyzed in sections 3 and 4. Section 6 contains the definition of the four dimensions and their related metrics (or degrees), which are used to compare the different SNA tools in Section 7. Finally, some challenges, future trends in this area, and the main conclusions of this work are given in Section 8.

2 SNA Scientometric Analysis

Although bibliometrics emerged to support the daily work of librarians, nowadays it is used to evaluate the scientific achievements of people and institutions [22]. The main goal of bibliometrics is the quantification of scientific production, measuring the performance of institutions and people. One of the contributions of this paper is an updated review of the current state of the art in the area of SNA. Nevertheless, the number of research papers in this domain increases every day and it is near to impossible to cover in detail all the application domains in which SNA methods, algorithms, tools

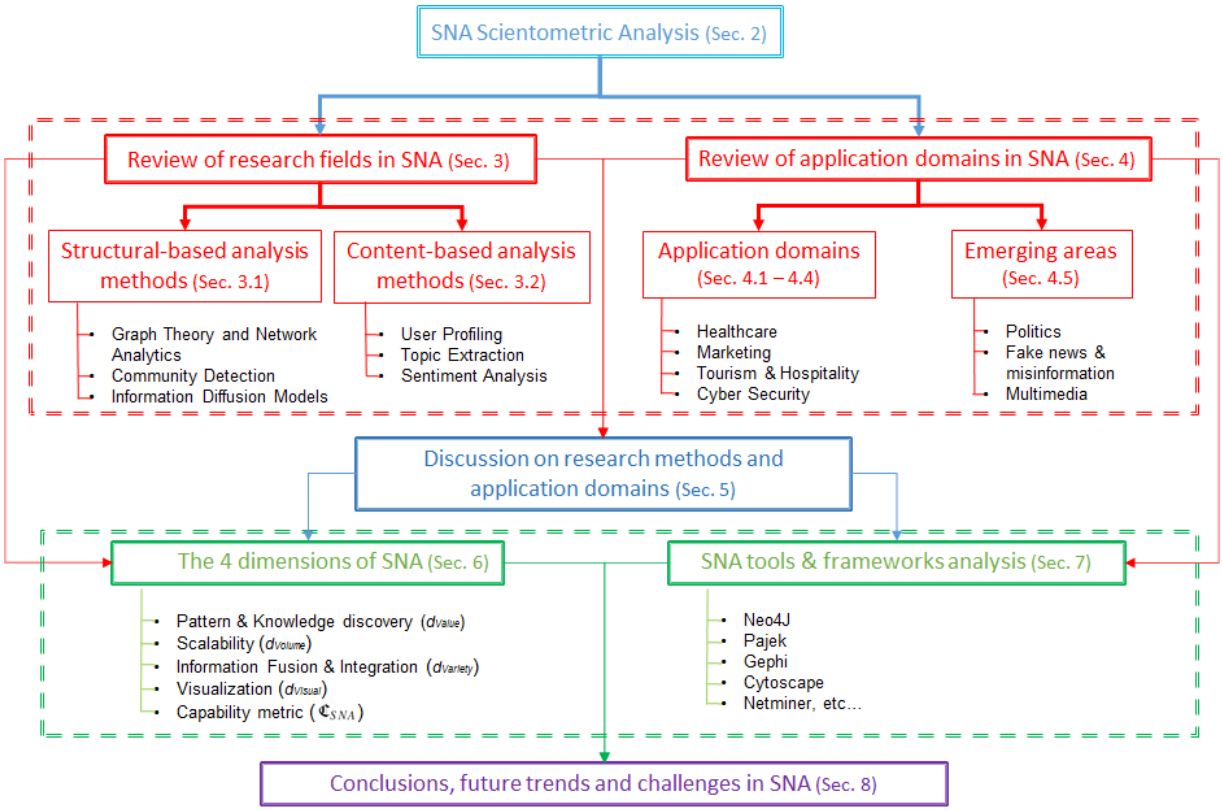


Figure 1: Structure of the paper and the main contents of different sections. Red boxes show the sections related to the review on both the current state of the art in SNA research, and their application domains, whereas green boxes show the contributions on the four dimensions defined, and the software tools and frameworks assessment carried out.

and frameworks are applied. This section presents a scientometric analysis of the research papers published in SNA, which has been carried out to detect and select those relevant areas that will be analyzed.

This study has been done by using the `Meta-knowledge Python` package [23], which accepts raw data from the Web of Science, Scopus, PubMed, ProQuest Dissertations and Theses, and select funding agencies as NSF (United States), or NSERC (Canada), among others. The output of this package is a set of characteristics for quantitative analysis, including Time Series methods, Standard and Multi Reference Publication Year Spectroscopy (RPYS), computational text analysis, and network analysis.

In particular, this analysis presents a review of works related to SNA using Web of Science as a search engine, covering the highly cited articles over a five years period (from 2014 to 2018) and resulting in a record collection of **28.805 articles**. It has only been considered in the scienciometric analysis those authors that have used in their publications the keyword (#hashtag) of *"#social network analysis (SNA)"*. Although this analysis could discard some relevant authors or publications, which did not use the previous term, this keyword was used to restrict the set of papers to analyze and to obtain some of the highly relevant application domains and research in SNA.

Firstly, to better understand the evolution in the area of SNA in the last sixty years (since the 70s), RPYS has been used. This method was proposed by Marx et al. [24], and it is a method for quantifying the impact of historical publications on research fields. Standard RPYS [25] analyzes the cited references and especially the referenced publication years of a publication collection. This method plots the cumulative distribution of cited references in terms of the referenced publication years. In the first step, all the references from the publications are selected (from a particular research field and period). Then the 5-year median deviations, to the number of cited references from each publication year, are computed to generate a *spectrogram*. The peaks in the spectrogram (deviations from the median) indicate those specific years with highly cited publications within the domain of the sample. Also, Multi RPYS is an extension of the standard method [25]. It segments the original citing articles based on their publication years and conducts a Standard RPYS

analyzes for each one, visualizing the results as a heat map. Therefore, this method is useful for differentiating between historical publications that have a lasting impact, versus those that are influential only within a short time frame.

As shown in Fig. 2, the years with pronounced peaks for SNA are: 1967 [26], 1973 [27], 1977 [28], 1979 [29], 1988 [30], 2011 [31] and 2012 [32]. Analyzing the most cited papers of this specific years, in the 70s and 80s, there were publications with a high impact on the topic, being published the articles most cited in journals in the area of Social Science [26, 27, 28, 29, 30]. While the most recent articles with higher impact are published in journals belonging to Computer Science [31, 32]. It can also be observed that in the 70s, 80s and 90s there were several publications with a high impact in the area of research, but from 1994 to 2010 there was a period of stability where no high relevant works emerged. Fig. 3 shows the Multi RPYS analysis, and the fact mentioned above can be seen. Between 1996 and 2010, there are publications for all the years analyzed that are quite quoted, but where no publication stands out from the rest. Therefore, it can be drawn that this period of years has very high activity in the research area of SNA, existing many publications in the topic which are very cited in general terms. Besides, according to the standard RPYS analysis, the oldest years that showed publications with a high impact, also appear as highlighted (1973, 1977, 1979 and 1988).

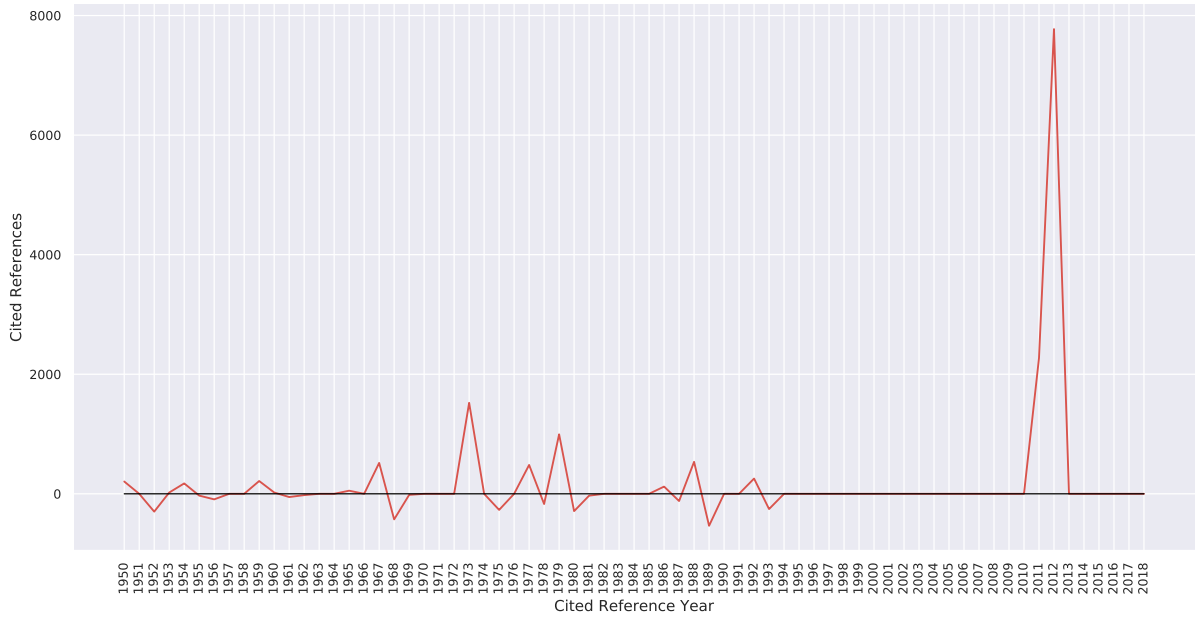


Figure 2: Standard RPYS related to SNA. Pronounced peaks represent years where citations to published books or articles deviate from a 5-year median.

From this long period analysis, it can be concluded that most of the papers published in the 70s and 80s were focused on defining the basic principles of the SNA such as the principle of diffusion of influence and information, mobility opportunity, community organization, or links joining people who know each other [26, 27, 30]. Also, published works at these years introduced a broad background for measures of structural centrality in OSN [28, 29]. However, since 2000, the main contributions have been focused on the application of these principles to specific domains, using OSN data to extract new knowledge, which improves the performance of the organizations. In this case, it can be shown how the areas of highly cited papers have been moved from Social Sciences (the 70s and 80s) [1, 27, 29, 30, 33], to Physics, Mathematics or Computer Science [34, 35, 36, 37]. These articles are mainly focused on the design of new methods, algorithms and applications for SNA, by taking into account the **structure** of the networks [34, 35] and detecting important structures, as communities, inside them [36, 37]. As it has been already seen in the RPYS analysis, the oldest publications that appear are related to the definition of principles and measures to model the dynamics of the OSNs. Whereas, the most recent publications are focused on the development of methods and techniques based on the concepts previously published. For example, it should be noted that community detection algorithms are those that prevail in the last two decades, such as Girvan and Newman [36, 38], edge betweenness centrality [39], fast Greedy [40, 41], Cfinder [42, 43], Walktrap [44, 45], structural algorithm by Rosvall et al. [46, 47], clique percolation method [48], fast modularity optimization by Blondel et al. [37], Louvain algorighm [49], community embeddings [50], etc.

To identify the most relevant topics related to SNA, textual analysis has been performed using the collection of articles gathered from the period of the last 5 years studied. In this case, the latent Dirichlet allocation (LDA) model has been

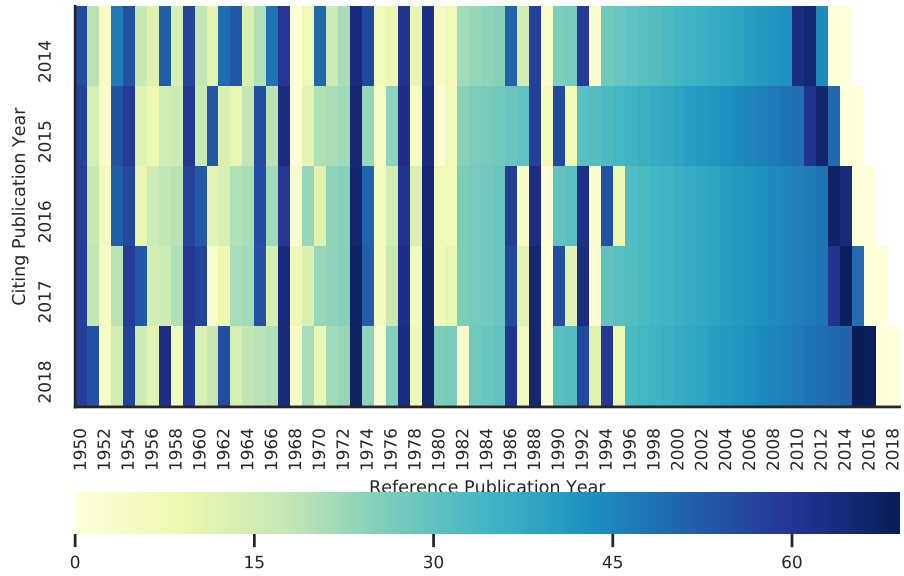


Figure 3: Heatmap showing the results of a Multi RPYS analysis. Darker bands across both periods indicate lasting influence.

applied to detect the top 20 topics, by processing the “keywords" used in the article collection, to later visualize the most frequent terms (i.e., keywords) as a world cloud (see Fig. 4).

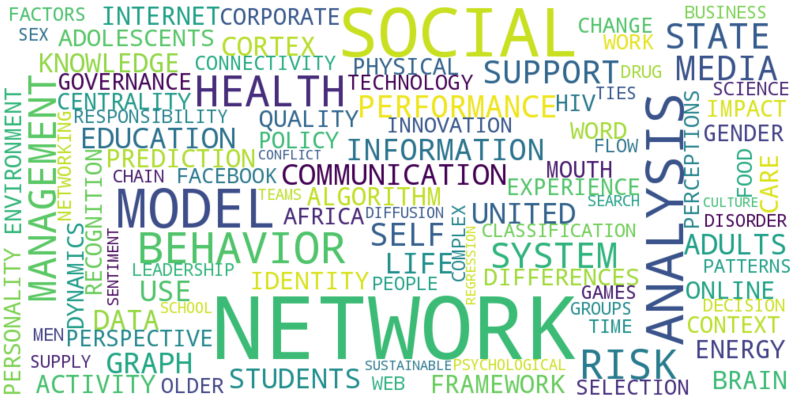


Figure 4: World Cloud visualization of the most frequent terms that appear in the top 20 of extracted topics processing the Keywords of the papers.

Analyzing the most relevant keywords related to SNA, it can be seen at first sight that many of the relevant keywords are related to the key topic of *Pattern & Knowledge Discovery* are: Model, behavior, Patterns, Classification, Prediction, Knowledge, Regression, Recognition, Dynamics, Groups, Diffusion, or Sentiment, among others. Also, there are several relevant application domains where these methods of knowledge extraction are applied as Healthcare (HIV, Disorder, Brain, Cortex, Drug, Sex, Physical and Psychological), Education, Governance, Policy, Mouth (Mouth-to-Mouth), Business, Food, Culture, or Games. Regarding the topic of *Information Fusion*, there are several keywords related to different data sources and types of information that are used in the area such as Internet, Media, Facebook, Web, Graph, Factors, Activity or Context. The rest of the keywords would be classified taking into account the topic of *Scalability*, where the terms Performance, Complex and Sustainable keywords are directly related to this feature. And finally, related to the topic of *Visualization*, appears some keywords such as Perspective, States, or Flow.

Although the initial study covers the last six decades, a detailed analysis of the last five years has been done to understand which areas and application are more relevant in terms of high impact. From this analysis, it can be concluded that

currently some of the hot research areas, from Computer Science, in this topic are: Data Science and Big Data, more specifically Network analysis, Social Media, Sentiment analysis, Text Mining, and Information diffusion. Whereas some of the hot application areas are: Health, Marketing and Business, and Tourism. It is particularly important to point out that a large number of published works are focused on solving problems (through the design of specific algorithms and techniques, such as those related to community finding problems, or information diffusion models), which are related to the problems of pattern mining and knowledge discovery, how to fusion or integrate information, how to visualize adequately the information, or how to handle huge amounts of data.

Finally, and although the scientometric analysis has not directly shed light on some areas such as fake news and misinformation (e.g., fake news analysis in SN), or cybersecurity (e.g., cyber intelligence, cybercrime, and cyber terrorism), during our analysis of the state of the art we have detected an increasing research activity in these areas (see Section 4.5 for a further description), so they will be analyzed as emerging areas due to its high potential in the near future. Previous findings related to the research and application domains in SNA have guided the final decision on the selection of what fields will be finally analyzed in this paper.

3 Techniques and algorithms

As it has been shown in the previous section, the area of SNA comprises a wide variety of multidisciplinary fields, ranging from Social Sciences to Sciences and Engineering. This section is focused on current findings from the Computer Science point of view. Currently, in Computer Science, and other related areas such as Mathematics or Physics, the design and development of new algorithms to process, extract and visualize (huge amount of) knowledge, are closely associated to *Data Science* and *Big Data* fields. The former can be defined generically, as a multi-disciplinary field that uses scientific methods, techniques and algorithms, to extract useful knowledge from structured and unstructured data. Whereas the latter uses different methods and techniques to analyze, and systematically extract information from data sets that are too large, or complex, to be dealt with traditional data-processing algorithms. From the data engineering perspective, an OSN can be analyzed from two different points of views [6]: (1) *structural data* that represents the connections, interactions (linkage-based), and the topology of the network; or as (2) *content data* that is focused on the information contained and shared in the OSN by the users. The isolated analysis of one of these two types of data would provide an incomplete vision of the information stored in the network, so the underlying patterns and knowledge could be lost. Therefore, and taking into account these two types of data, we can distinguish between two different approaches: *Structural-based analysis* [6, 51] and *Content-based analysis* [6, 52], that are described below.

3.1 Structural-based Analysis

This is the research area involved in the analysis of networks using network theory (usually known as graph theory) [5, 53, 54]. Their main methods and techniques have been applied in large network-based problems as OSNs and social media applications (i.e., Facebook, Instagram, Twitter, WhatsApp, LinkedIn, Snapchat, Youtube, Tumblr, Pinterest, Skype, or WeChat to mention just a few) [55, 56]. OSNs are considered complex networks [57] because they present non-trivial topological features. In other words, the connection patterns between nodes will not be random or purely regular [58]. This section provides a short introduction to some classical, but essential, concepts on graph theory and network analysis in SNA, and their particular application to structural-based problems in OSNs, such as community finding or information diffusion.

3.1.1 Graph Theory and Network Analytics

SNA examines the structure of relationships between social entities. For example, in mathematics and computer science, graph theory represents social networks as a graph, through nodes connected by edges, where the nodes would be the individuals and the edges would be the relationships that joint them. Nodes and edges compose the graph, a data structure that allows describing the properties of the social network. Graph theory [59, 60] is an area of an important contribution to research in data analysis. Through this theory, it is possible to generalize and analyze the existing social interactions between users, and what is their behavior within their contact network [61, 62, 63]. Currently, the study, analysis and visualization of social networks focus mainly on the association and measurement of interactions between people, groups, organizations, or any set of entities with a relationship between them. From Graph theory, concepts as *betweenness, clique, centrality, cohesion, component, connectivity, degree, diameter, path, transitivity*, and others are used to represent the network structure and topology [5, 6, 51, 56, 53, 29]. Therefore, and from a merely technical point of view, SNA provides both visualization [20, 64, 65] and mathematical tools [51, 66] that allow the analysis and study of non-trivial relationships (patterns, structures, etc.) that can be represented within a network. Following there is a brief description of some basics concepts on Graph Theory and Network analysis [59, 60, 67]:

- **Types of graphs**: a graph $G = (V, E)$ can be defined as a set of vertices, or nodes, denoted by $V = \{v_1, \ldots, v_n\}$, and a set of edges E, where each edge is denoted by e_{ij}, if there is a connection between the vertices v_i and v_j, with $v_i, v_j \in V$. Graphs can be *directed* or *undirected*. The graph will be directed if $e_{ij} \neq e_{ji},\ \forall i, j$, otherwise the graph is undirected. In some cases, edges have associated one, or more numerical values, called *weights*. In this case, G is a *weighted graph* if exists a function $w : E \rightarrow R$ which assigns a real value to each edge ($G(V, E, W)$). In other cases, these attributes can be categorical [68, 69], or simply signed [70, 71, 72]. There are a large number of different types of graphs that can be found in the literature, such as: trees and forests (a tree is a graph structure that has not cycles between nodes, a graph that can be represented as a set of disconnected trees is called a forest), spanning trees (for any connected graph is a tree that includes all the nodes of the graph, but not all the edges. The edges included will be those necessary so any two pair of nodes of the tree should be connected. Therefore, for any graph it would be possible to find multiple spanning trees, if a weighted graph is considered, the one with the minimum weight is called the *Minimum Spanning Tree*: MST). Other interesting graphs are Steiner trees, Complete graphs, Planar graphs, Bipartite graphs, Regular graphs, or Bridges among many others [59, 67, 73].
- **Graph & Network models**: graph models are one of the most powerful and useful tools to represent diverse types of data. This kind of models is particularly suitable for network-based domains and problems. Any network (or graph) model has as the main goal to reproduce or mimic the main characteristics from the real network considered. To determine these characteristics, several measures are identified and assessed to ensure that these measurements are consistent with a real-world network. Three network attributes are commonly used to measure this behavior: *degree distribution*, *clustering coefficient*, and *average path length* [59, 73]. Where degree distribution denotes how node degrees are distributed across a network. The clustering coefficient measures the transitivity of a network. And finally, the average path length denotes the average distance (shortest path length) between any pairs of nodes. The fundamental graph or network models which can be found in the literature are the following:
  - *Real-World networks* [74]. One of the basic characteristics shared by this type of networks is the fact that connections (or ties) between nodes depends on a specific probability. In real-world networks, such as social networks, this means that the probability of a tie between two nodes is higher if there exists any kind of relationship (friendship, professional, etc.) between them. The degree distribution values in a real-world network, such as social networks, follow a *power-law distribution*. However, as it was demonstrated in [74], social networks differ from other types of networks, mainly because social networks are more easily divided into communities than non-social networks. Therefore, this characteristic affects the degree distribution, usually with degrees being positively correlated, and a high level of clustering.
  - *Random networks* [75, 76]. From a theoretical perspective, random networks, most commonly known as random graphs, belong to the intersection between graph theory and probability theory [75]. Any random graph can be described as a probability distribution, or as a random process, which will be used to generate the connections between any pair of nodes in the graph. Therefore, to generate a random graph, and starting from a set of n isolated nodes, it is only necessary to randomly add (following a predefined probability distribution) successive edges between any pair of nodes. Random graphs are commonly used to compare the structure and properties of any graph. The two most relevant models to generate random graphs are the Erdös-Rényi model [77], and the one proposed by Gilbert [78]. The main difference between both models lies in how the connection probability (i.e., the probability distribution) is managed by each model. In Erdös-Rényi model, all nodes and edges share the same probability. This means that in this model, two nodes are chosen uniformly random and the link connecting them is created, whereas, when the Gilbert model is used an independent probability for each edge in the graph is defined.
  - *Small-World networks* [35]. In many real world networks, there are two basic properties: the first one is related to the distance between two nodes (which it is usually small) whereas the second one is related to the transitivity, or clustering coefficient of these networks (which is usually relatively high). Erdös-Rényi random networks usually have a low average path length, so it is possible to move from one node to other in the graph with a few numbers of hops. This property, which is shared by a large number of real-world networks, is often called the *small world property*. This concept was popularized by terms like the 'six degrees of separation' between two people, meaning that any two persons are distanced by at most six friendship links. The concept of six degrees of separation comes from the well-known Stanley Milgram experiment. This famed social psychologist, conducted an experiment in 1967, which tracked chains of acquaintances in the United States. In this experiment, people were requested to route packages to a fixed recipient (a stockbroker from Boston, Massachusetts), passing them only through their direct acquaintances. In this experiment, Milgram found that the average number of people needed to send the packages between two the original sender and the destination recipient were five (i.e., six degrees of separation) [26]. However, when real-world networks, such as those related to social networks, are

analyzed, it is quite common to find one property that does not appear at Erdös-Rényi random graphs. This property is related to the transitivity degree, which in these real networks have a higher value than in Erdös-Rényi random graphs. This property appears due to the social behavior of the network (usually your friends are likely to be friends), which means that the number of triangles (i.e., transitivity in the graph, or the clustering coefficient value) will have a larger value in social networks.

- *Scale-free networks* [79]. This kind of networks is commonly defined based on their degree distribution. In these networks, the degree distribution of nodes will follow a *power law* (at least asymptotically). That is, the fraction $P(k)$ of nodes in the network having k connections to other nodes goes for large values of k as $P(k) \sim k^{-\gamma}$, where γ is a parameter whose value is typically in the range $2 < \gamma < 3$, although occasionally it may lie outside these bounds [80]. There exists a variety of scale-free network-modeling algorithms, a well established one is the model proposed by Barabasi and Albert [34], usually called *preferential attachment* [81], or the Barabasi-Albert (BA) model, and it is defined as follows: "When new nodes are added to networks, they are more likely to connect to existing nodes that many others have connected to". As it was demonstrated by authors, several natural and human-made systems (such as Internet, the World Wide Web, citation networks, and some OSNs) are thought to be approximately scale-free and contain few nodes (named *hubs*) with unusually high degree as compared to the other nodes of the network (e.g., a singer, an athlete, a film actor/actress, or the extremely popular, influencers, youtubers, gamers, ...).

- **Network metrics**: Once, the most general and simple concepts, and models, from graph theory have been introduced, we can proceed with the definition of some basic metrics, or measures, that are used by graph algorithms. There are excellent books and references to these algorithms, which can be consulted to better understand how they have been designed, and applied to the domain of network analysis [6, 59, 73, 82, 83, 84, 85].
  - **Centrality**. Centrality is one of the essential metrics in graph and network theory, this metric is used to asses the relevance, or structural importance, of a node in the network. The centrality measure defines how important a node is in a network. In OSNs, this measure can be used to detect or identify, the most influential people in the network. When centrality is assessed, several measures are used: (1) *Degree Centrality*, which ranks nodes with more connections higher in terms of centrality; (2) *Eigenvector Centrality* tries to generalize the degree centrality by incorporating the importance of the neighbors (in directed graphs it can be used incoming or outgoing neighbors); (3) *PageRank* measure takes into account the value of passed centrality by the number of outgoing links (outdegree) from that node, so this measure gets a fraction of the centrality values of the nodes connected to the node from the source node considered; (4) *Betweenness Centrality* computes the number of shortest paths that traverse any two nodes in the graph, v_i and v_j, nodes with higher betweenness values can be seen as "bridges" between different subgraphs, this measure is used by some specific community finding algorithms; (5) *Closeness Centrality*: the idea of this measure is that the more central the nodes are, the easier will be to reach other nodes. Therefore, the smaller the average shortest path length is, the centrality value of the node will be higher; and (6) *Group Centralities* such as Group Degree Centrality, Group Betweenness Centrality, Group Closeness Centrality that generalize the previous centrality-based measures to a group of nodes.
  - **Transitivity** and **reciprocity** are used to represent *linking behavior* in a network. Transitivity analyzes the linking behavior to determine whether it demonstrates a transitive behavior between three nodes, so at least three edges will be needed to create a triangle. Higher transitivity on a graph results in a denser graph, which in turn is closer to a complete graph. Therefore, it is possible to determine how close graphs are to the complete graph by measuring the transitivity. This can be performed by measuring the *[global] clustering coefficient* and *local clustering coefficient*. The former is computed for the network, whereas the latter is computed for a node.
  - **Balance** and **status**. A signed graph is a graph in which each edge has a positive or negative sign. This sign is used in OSNs to represent interpersonal relationships (e.g., such as friends or foes, boss or subordinate, social status). This kind of graph is *balanced* if the product of edge signs around every cycle is positive. In real-world social networks, we expect some level of consistency concerning these interactions. For example, it is more plausible for a friend of one's friend to be a friend than to be an enemy. In signed graphs, this consistency translates to observe cycles (triads, triangles) with three positive edges (i.e., all friends) more frequently than the ones with two positive edges and one negative edge (i.e., a friend's friend is an enemy). *Social balance* and *social status* are used to determine the consistency in signed networks. Social balance theory says that friend/foe relationships are consistent when the transitivity between nodes can be propagated, such as "The friend of my friend is my friend". Therefore, triangles that are consistently based on this theory are denoted as balanced (whereas, inconsistent triangles are denoted as unbalanced), in these balanced triangles, an even number of negative edges will appear.

Social status theory measures how consistent individuals are in assigning status to their neighbors. The idea is simple, and it can be summarized that if a person X has a higher status than Y, and this last person has a higher status than Z, then X should have a higher status than Z. In a signed (directed) graph representation, positive and negative signs will be used to show higher or lower status depending on the arrow direction (if the edge has a positive value, it will mean than outgoing node has a higher status than incoming node, if the edge has a negative value, it will represent the opposite).

- **Graph algorithms**: from the graph theory area, there are a large number of algorithms that have been designed to work with graphs to compute certain aspects of them (paths, flows, bridge, etc.). Some of the most frequently used in the area of SNA are: Graph/Tree Traversal, Shortest Path Algorithms, Minimum Spanning Trees, Network Flow Algorithms, Maximum Bipartite Matching, and Bridge Detection, among many others [86].

3.1.2 Community Detection

Community detection problem (CDP) can be defined as the division of the graph into clusters of nodes based on the network structure [87]. The main idea behind CDP is that nodes belonging to the same cluster are strongly interconnected whereas they maintain sparse connections to the nodes of other clusters. This problem is similar to the idea of graph partitioning into groups of nodes according to the network topology, where a partition is a division of the graph and it can be easily mapped into a cluster [41, 88, 89, 90].

To detect the communities, or clusters, on a graph, there is a wide range of techniques such as random walks, spectral clustering, modularity maximization, or statistical approaches [88]. This kind of algorithms uses the topology of the graph to create the partitions that are validated by taking into account the density of the resulting sub-graph (i.e., a sub-graph is highly connected), and connections from these nodes to the rest. A good community is the one whose nodes are highly connected and it has few connections to the nodes of other communities [91]. Considering both connectivity and density, a possible definition of a graph cluster could be a connected component or a maximal clique [92]. This is a sub-graph into which no vertex can be added without losing the clique property (a clique, in an undirected graph, is a subset of the vertices, such that every two distinct vertices are adjacent).

However, it is not always clear that a vertex should be assigned only to a unique cluster, or to several. For this reason, there are several taxonomies to distinguish the different algorithms and approaches developed to solve CDP, but the two most extended taxonomies are: (1) the one that pay attention to the number of communities that a single node can belong (at the same time) and (2) the second one, that pays attention to the type of analysis (static vs. dynamic) performed in the graph.

The first taxonomy refers to *non-overlapping*, or *overlapping* algorithms depending on the number of clusters that any node can belong at the same time. In this sense, *non-overlapping* methods are those whose nodes belong only to one community, whereas in *overlapping* approaches the nodes may belong to several communities at the same time.

- *Non-overlapping community finding algorithms*. One of the most well-known algorithms for community detection was proposed by Girvan and Newman [36]. This method is based on the "edge betweenness" metric that measures the number of shortest paths that go through a given edge. Using this metric it is possible to isolate the different communities by removing those edges with higher betweenness (these edges can be understood as "bridges" connecting two different communities). The main drawback of this algorithm is its high complexity, and its computational costs if it is applied to large networks.

  Another very popular metric to compute communities is called *Modularity*. Despite the computation of this metric is an NP-complete problem, there is a high number of algorithms able to use it to detect communities in a reasonable computational time. One of these algorithms is the one proposed by Newman [38]. This greedy algorithm is based on an agglomerative hierarchical clustering where the groups of nodes are successively joined together to create larger communities with higher modularity. Due to the sparseness of the adjacency matrix, this algorithm involves a high number of operations when this matrix is updated. In order to reduce these operations, Clauset et al. [40] proposed an algorithm that uses the matrix of modularity variations to improve the performance.

  Newman reformulated the modularity measure in terms of eigenvectors by replacing the Laplacian matrix with the modularity matrix [93], called the spectral optimization of modularity. This improvement has also been used to improve the results of other optimization techniques [94, 95].

  As has been stated, a good community is the one with high density between the nodes belonging to the community. Therefore, a random walker would spend a lot of time inside a community because there would be a high number of paths that can be followed. This is the idea that relies on behind the *random walk* algorithms [44]. Zhou and Lipowsky [96], based on the fact that walkers move preferentially towards vertices that share a large number of neighbors, defined a proximity index that indicates how close a pair of vertices is

to the rest of vertices. Communities are detected with a procedure called NetWalk, which is an agglomerative hierarchical clustering method by which the similarity between vertices is expressed by their proximity.

- *Overlapping community finding algorithms* are those algorithms that rely on the idea that nodes can belong to several clusters at the same time. This idea can be found in real-world networks, for example, people in an OSN can belong to different groups or communities. To represent this feature in the algorithms, fuzzy clustering algorithms applied to graphs [97] and overlapping approaches [89] have been proposed.

  A good review of the state of the art in overlapping community finding algorithm was carried out by Xie et al. [87]. This work noticed that for low overlapping density networks, SLPA, OSLOM, Game, and COPRA offer better performance. For networks with high overlapping density and high overlapping diversity, both SLPA and Game provide relatively stable performance. However, test results also suggested that the detection in such networks is still not yet fully solved. A common feature that is observed by several algorithms in real-world networks is the relatively small fraction of overlapping nodes (typically less than 30%), each of which belongs to only 2 or 3 communities. Other approaches based on genetic algorithms, multi-objective GA, and evolutionary strategies can be found at [89, 98, 99].

The second taxonomy, which can be used to categorize the different algorithms and approaches is the one that takes into account how the variable '*time*' is integrated into the model. In this sense, it is possible to talk about *static* and *dynamic* community finding algorithms.

- *Static community finding algorithms* refers to those algorithms and approaches that do not take into account the evolution of the OSN, i.e., the variable '*time*' is not modeled into the system. The algorithms previously described are static, which means that the network is modeled into a single snapshot that contains all the information regarding the OSN. These algorithms present some advantages and drawbacks. On the one hand, static community finding algorithms are quite easy to apply to any problem, because no changes will occur in the network during the algorithm execution. On the other hand, and as one of the drawbacks, this kind of non-temporal models makes that the results may not be very representative, because OSNs are continuously changing due to the high variability in the number of users (nodes) and interactions (edges) between them.

  The different algorithms that belong to the static community finding problem can be grouped in four different categories depending on the scope of the corresponding algorithm. In this sense, there are node-centric, group-centric, network-centric and hierarchy-centric algorithms [100]:

  - Node-centric community finding methods: in this case, each node of the network must satisfy the properties of mutuality, reachability and degrees.
    - *Mutuality property* relies on the concept of the clique, which is the maximal complete subgraph of three or more nodes in such a way all of them are adjacent to each other. Finding the maximum clique in a network is an NP-hard problem, for this reason, it is quite popular to develop algorithms able to find approximate solutions. One of these algorithms is the one proposed in [101], where each time a subset of the network is analyzed. In this algorithm, a greedy-search procedure is executed to find the different cliques in each subnetwork. Then, the maximal clique found is used as a lower bound in a pruning process, i.e., if the maximal clique contains x nodes, those nodes whose degree is lower than x can be removed. This process is repeated until the network is simplified to a reasonable size.
    - *Reachability property* considers that two nodes may belong to the same community if there is a path connecting them. This property assigns the nodes belonging to the same connected component to the same community. The advantage of this property is that it can be computed in $\mathcal{O}(n+m)$ time, but real-world networks are composed by a big component whereas the majority are singletons and small communities [102]. For this reason, the identification of communities in the small components is straightforward but some efforts must be done to detect the communities contained in the biggest connected components. One way to do that is by finding the *k-cliques*. A k-clique is a maximal subgraph where the largest geodesic distance between two nodes is less or equal to *k*, i.e., $d(i,j) \leq k \; \forall v_i, v_j \in V_S$, where V_S is the set of nodes in the subgraph S.
    - *Nodal degree property* establishes that the nodes of a group must be adjacent to a relatively large number of group members. In this case there are two different structures studied: *k-plex* [103] and *k-core* [104]. The former is a subgraph composed of n_s nodes and each node is adjacent to no less than $n_s - k$ nodes in the subgraph. The latter is a subgraph composed of n_s nodes and each node is connected to at least k members.
  - Group-centric community finding methods. In this category falls all the algorithms and methods, that consider the connections inside the community as a whole. These algorithms are usually known as *density-based* algorithms and are based on the concept of γ-dense subgraphs, or quasi-clique [101]. A subgraph $G_S = (V_S, E_S)$ is γ-dense if:

$$\frac{E_S}{V_S(V_S-1)/2} \geq \gamma \tag{1}$$

The work performed by Abello et al. [101] uses a GRASP algorithm to find a maximal quasi-clique. This procedure starts the quasi-clique with the vertex with the largest degree in the network. Then, iteratively, the quasi-clique is expanded with those nodes that are likely to contribute to a larger quasi-clique.

- Network-centric community finding methods consider all the connections of the network, instead of only the connections of the community as Group-centric methods do. In this type of algorithms, the different nodes are grouped in sets of disjoint communities according to a specific quantitative criterion.
  - *Group based on Minimum-Cut*. According to this criteria, a community is defined as a subset of nodes $C \subset V$, such that $\forall v \in C$, v has at least as many edges connecting to nodes in the same community as it does to vertices in $V \setminus C$. In [105] authors showed that the community can be found via $s-t$ minimum cut, where s is the source node in the community and t is the sink node outside the community.
  - *Group based on Modularity*. In this case, the structure of the community is compared against a random graph, more precisely the modularity defines how likely the community structure is created at random. Considering that A is the adjacency matrix, s_i denotes the community that node v_i belongs to, and d_i represents the degree of vertex i, the modularity is computed as follows:

$$Q = \frac{1}{2m}\sum_{ij}\left[A_{ij} - \frac{d_i d_j}{2m}\right]\delta(s_i, s_j) \tag{2}$$

    where $\delta(s_i, s_j) = 1$ if $s_i = s_j$. In general, the goal is to find those communities that maximize Q, but a negative value for the modularity indicates that vertices are assigned to bad communities.
  - *Group based on Latent Space Model*. The idea is to build a latent space [106, 107] with the nodes of the network in such a way those nodes with dense connections are close to each other. These models assume that the interactions between nodes depend on the position of the nodes in the latent space.
- Hierarchy-centric community finding methods try to build a hierarchical structure of communities by taking into account the structure of the network. These methods are quite similar to those from hierarchical clustering research field [108, 109]
  - *Divisive hierarchical clustering* starts dividing the whole network into several disjoint sets, and then each set is split into smaller ones until all subsets contain only one node. A popular algorithm in divisive hierarchical clustering is the one based on edge betweenness [39]. In this case, the algorithm removes those edges with higher betweenness in such a way the different communities are isolated.
  - *Agglomerative hierarchical clustering* corresponds to the opposite approach. In this case, the algorithms and methods start assigning each node to one independent cluster. In each iteration, clusters are merged into a larger one according to a specific metric. The most popular metric used in this type of algorithms is modularity [40]. The idea is to merge two communities if the resulting community improves the modularity. The process is repeated until there are no changes in the modularity. The main problem with this approach is that it generates many imbalanced merges with result in high computational cost [16].

- *Dynamic community finding algorithms* refer to those approaches that incorporate the variable *time* into the model. Working with time broadens the range of possible analysis. When time is involved in the model, not only the community structure of the network at any given time can be analyzed, but also their dynamics, i.e., their evolution over time. Regarding community dynamics, one could be interested in analyzing the life cycle of a community, when it appears for the first time, when it grows, when it splits into several communities ...etc. Likewise, one could be interested in finding communities that persist over time or dividing a network into periods where the community structure is stable. This variety of analysis, with different outputs and goals, are all encompassed inside the dynamic community finding literature.

  Embedding the 'time' variable into a network model is not a trivial task. Rossetti and Cazabet presented in [110] several models of growing complexity and precision. The first, and simplest model, is the *Aggregate* one. This model consists of generating a weighted graph, where the weights represent the number of times an interaction has occurred. Even though the *Aggregate* model allows doing some types of dynamic analysis, like tie strength estimation, it is very limited and does not capture the network dynamics. Consequently, the *Snapshot* and *Temporal Networks* models were proposed. The former introduces dynamic into a network by generating an ordered sequence of graphs, where each graph represents the state of the network at a given

point in time. The latter avoid doing any aggregation at all, and represents the network as a set of timestamped nodes and edges that precisely define when an element appear and disappear from a network. These models allow more expressivity at the expense of requiring more complex analysis.

Using both the *Snaphsot* and the *Temporal Networks* models entails a series of new challenges [110, 111]. For instance, when using the *Snapshot* model, the granularity of the network (the frequency at which the snapshot are taken) need to be decided. This is not a trivial task and depends on the context of the problem. Selecting an appropriate granularity is of utmost importance as the selection of this threshold can drastically affect the results obtained. Besides, the communities found on adjacent snapshots of the network need to be related to each other to be able to analyze their dynamics (growth, merge, ... etc). Finally, another challenge related to community finding that must be addressed is the problem of instability. Usually, community detection algorithms are unstable, many of the efficient community detection algorithms are stochastic, when we observe an event between two snapshots, we cannot be sure if they correspond to real events or an effect due to the stochastic nature of community finding methods. However, when using the *Temporal Networks* model we face similar problems. For instance, community events (growth, merge, ... etc) occurring during the evolution of the network still need to be identified. Moreover, it is difficult to ensure long-term coherence when using *Temporal Networks*. Due to the nature of the *Temporal Network* model, communities are continuously being updated over time. In the long run, this could lead to significant differences in the communities found compared to the static state of the network at a given point in time.

Several taxonomies have been proposed in the literature to categorize dynamic community finding methods. However, in this work, we will use a different one. First, the methods will be categorized according to the model used. Then, these methods will be further categorized according to the analysis followed. Following this taxonomy, we can divide the methods in those that work with *Snapshots* and the ones that work with *Temporal Networks*. As aforementioned, each category will be further divided into more sub-categories depending on their end goal. On the one hand, methods in the snapshot category can be further subdivided into four subcategories (Snapshot Community Tracking, Snapshot Community Detection, Consensus Community Detection, and Change Point Detection). On the other hand, methods in the temporal network category can be further subdivided into three subcategories (Community Structure Update, Temporal Community Tracking, and Persistent Community Finding).

- Snapshots-based dynamic community finding methods:
  - *Snapshot Community Tracking*: methods in this subcategory try to characterize a community life cycle over the network evolution. They split the community detection method into two steps: (1) uncover communities in each snapshot using some other method; (2) track communities among consecutive snapshots to define their life cycle. To do so, Palla et al. initially defined the dynamics of a community using six operations (Growth, Contraction, Merging, Splitting, Birth, and Death) [43]. Those operations were lately extended by Cazabet et al. with the addition of a 'Resurgence' operation [112]. These methods characterize a community as a sequence of consecutive events that describe what happens between consecutive snapshots. Some relevant works in this category are [43, 113, 114, 115, 116, 117, 118].
  - *Snapshot Community Detection*: these methods are focused on finding suitable communities structure for each snapshot of the network by processing the snapshots in their natural order using previously found results to guide the procedure. While some methods in this subcategory try to stabilize the community structure over time [119], others try to reduce the computational effort required using evolutionary strategies [120, 121] or combining bio-inspired meta-heuristics and novelty search strategies [122]. Some relevant works in this category are [119, 123, 124, 125, 126, 127, 128, 129].
  - *Consensus Community Detection*: methods in this category focus on finding one community structure that fits all the snapshots of the network. These methods process all the snapshots of the network simultaneously in a single process finding communities not only composed of nodes from the same snapshot but also composed of nodes from different ones. Some relevant works in this category are [130, 131, 132, 133, 134, 135].
  - *Change Point Detection*: the methods in this subcategory are related to the change point detection problem. They focus on splitting a network into different homogeneous periods separated by dramatic changes while also finding a suitable community structure for each period. Some relevant works in this category are [136, 137, 138, 139, 140, 141].
- Temporal Networks-based dynamic community finding methods:
  - *Community Structure Update*: the methods in this subcategory track community structures in an iterative way. Each change of the network is processed in a streaming way updating the actual community structure when a change demands it. This subcategory includes methods that define a set

of rules to update a community given a change or methods that do dynamic optimization of some quality metric. Relevant works in this category are [142, 143, 144, 145, 146, 147].

* *Temporal Community Tracking*: the methods in this category adapt the community structure to the changes of the network while tracking events on the dynamic communities (birth, growth, split ...etc). Some relevant works in this category are [148, 149].
* *Persistence Community Finding*: the methods in this category try to find persistent structures in the whole network evolution. Some methods required a fixed duration to be used as threshold and others can find structures with arbitrary duration. Some relevant works in this category are [150, 151, 152, 153, 154, 155].

3.1.3 Information Diffusion Models

OSNs allow any user (u_i) to create new information in the network in such a way any user connected to u_i will receive this information (e.g., in Twitter any user can create a tweet, in Facebook create a new post, or upload a photo in Instagram). Nevertheless, a powerful tool of any OSN is that any user connected to u_i is able to send the received information to their own connections. For example, any user can 'retweet' tweets on Twitter, or 'share' a post on Facebook.

This property, along with the number of connected users to any OSN, makes that any content created may propagate fast on the network reaching a large number of potential readers. This content that is spread extremely fast, and it reaches a huge number of users in a short period, are commonly known as '*viral*' content of the OSN. Lots of users try to create viral content to gain popularity on the networks, and some others use this propagation procedure for marketing campaigns.

The classic example of marketing campaigns in OSNs is the study of the opinion of customers regarding new products [156, 157], but taking into account the propagation of information in OSNs, the goal is to design marketing campaigns (such as posts, tweets, videos or photos) that reach the larger number of possible users in the shorter period of time [158, 159]. Another application domain is politics, where different parties use OSN to promote their propaganda [160, 161, 162]. Nevertheless, in the last years, there has been a wide misuse of OSNs. In this sense, ISIS has used OSNs to spread their propaganda and to recruit new members [63, 163, 164, 165], or the propagation of fake news [166, 167].

Because of that, many researchers have focused their research on understanding how the information is spread in OSNs. More precisely, researchers have tried to define the different models that describe the diffusion process. In this sense, models can be grouped in two different categories: *explanatory* and *predictive* models [168]. On the one hand, explanatory models are based on the transmission of the epidemic, where there are users infected and users that are susceptible to be infected. On the other hand, predictive models are used to predict how the information will be spread through the network.

Explanatory models consider the diffusion process as an epidemic spread process, where the infection propagates between users in the same way as the information does. These type of models are based on the state of the different users and the models try to extract conclusion from how the users change their states. These states are the following:

- *Susceptible* (S): this state represents users that are not infected. In the analogy with the information diffusion, it means any user who has not received the information.
- *Infected* (I): it is used to represent those users infected by the virus or the users who have received the information.
- *Removed* (R): this state represents an infected user that that has been cured.

The classical models that belong to this category are susceptible to infected susceptible (SIS) [169], susceptible infected removed (SIR) [170] and susceptible infected removed susceptible (SIRS) [171]. All of them are based on the Susceptible-Infected model [172] that considers two states for the users (susceptible and infected), any susceptible user can be infected and once the user gets infected the state of the user cannot change. Based on this model, SIS, SIR and SIRS models differ in the number of states for each user and the transitions between these states.

The first model is SIS and it considers a daily rate of the cured patients, i.e., an infected user can be cured, and its state will change to Susceptible. The second model was proposed by Kermack and McKendrick in 1927. This model is called SIR, and it introduces a new state named *Removed* that represents those infected users that are cured. The difference between SIS and SIR is that in SIS models any cured infected user is susceptible again (which means that this user can be infected again), but in SIR models an infected user that is cured gets the state of *Removed*, which means that it is immune to the infection. Finally, the third model is called SIRS, and it considers that a cured user can become a susceptible user with a given probability. Fig. 5 shows a graphical comparison of the just explained models.

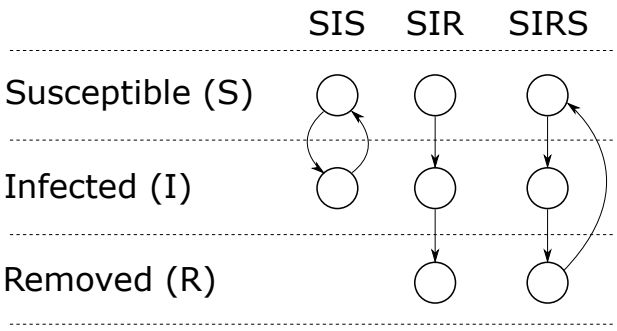


Figure 5: Comparison among the different main SI-based models: SIS, SIR, and SIRS.

The second category of diffusion processes are called predictive models, and its most popular, and well-known models are: the independent cascade model (ICM) [173] and the linear threshold model (LTM). The goal of these models is to predict the future information diffusion process.

On the one hand, ICM was proposed by Goldenberg et al. [174, 175] and it is inspired by the theory of interacting particle systems. This model takes into account the probability that an active user infects an inactive one. In this way, given two nodes that are connected in the network: u and i where u is active and i is inactive, the probability of u to infect i will be denoted as $P(u, i)$. There are two hypotheses in this model: the first one is that the probability of a node u to infect node i is independent of the influence of other active nodes connected to i. The second hypothesis is that any active node u has only one chance to infect i, independently on the result (success or not) node u will never try to infect i.

On the other hand, LTM was proposed by Watts [176] and the individuals make a decision based on its neighbors. In this model each individual has a state (*active* or *inactive*) and a threshold ($\phi \in [0, 1]$) that will define the activation level of the individual. Initially, there are a small set of users that are active. At each step, if the fraction of active neighbors of a given inactive individual is greater than its threshold, the given individual will change to active. The diffusion process finishes when the number of active individuals becomes stable.

3.2 Content-based Analysis

As it has happened since the origin of the Web, the content published by humans is easily understandable by them, however and due to the lack of structure and the multimodality of the information (text, images, video, audio), the automatic analysis of the information has been one of the biggest challenges for those algorithms that must gather and extract knowledge from this data [18]. Both facts, the lack of structure and multimodality of the information, affect the analysis of information contained in OSNs. This content analysis requires computational methods that transform unstructured content into structured information. One of the most successful areas of research in this context is natural language processing (NLP) [177]. NLP provides a set of methods and algorithms that enable the processing of multimodal information circulating on OSNs, thus allowing unstructured information to be transformed into structured information. Although, other approaches have been used to analyze video and images from OSNs, these methods can be regarded as immature when are compared against NLP [178, 179]. Other successful areas, which have been extensively used to gather knowledge from OSNs, have been Data Science [56] and Big Data [180, 181]. Their main methods and algorithms related to data collection, cleaning, pre-processing or mining, have been used to gather, model and extract patterns from OSNs [17, 182, 183]. The most representative applications of OSN content analysis are user profiling (Section 3.2.1), topic extraction (Section 3.2.2) and sentiment analysis (Section 3.2.3).

3.2.1 User Profiling

Content-based SNA mostly focuses on the contents of the interaction between nodes in an OSN to extract topics or opinions (see next sections). Content-based SNA, however, can also be about the information about the nodes themselves, i.e., user profiling. User profiles are established based on the behavioral patterns, correlations and activities of the user analyzed from the aggregated data using techniques like clustering, behavioral analysis, content analysis and face detection (the mechanisms used in profiling users vary depending on application and purpose).

Profiling user in OSNs requires data pertaining to the user and the online activities of such user within the OSN. Those activities may depend on one's interest or they can be the effect of some influence on them. Pal and McCallum [184] exploited the content of communication messages between users to cluster email recipients into groups. For each user,

they built a model that maps keywords and phrases extracted from email messages to the recipients who are likely to receive an email containing those terms. Bar-Yossef et al. [185] and Roth et al. [186] observed how users group their friends when sending email messages. Based on past behavior of user's group communication messages, they developed ranking algorithms that can predict other similar users who can belong to a particular group specified as a seed-set of users. De Choudhury et al. [187] proposed an approach to 'label' nodes in an OSN as per their roles, e.g., 'student', 'faculty', or 'director'. They applied their approach on an email communication graph, filtering out infrequent email communications below a certain threshold.

Cai et al. [188] proposed a model for mining and identifying the top-k influential bloggers based on parameters such as comments, domain of interest and page link network authority. The developed model can be used for multiple application scenarios. Sun et al. [189] proposed a novel algorithm to recommend influential bloggers based on the observation that the reproduction of blogposts and similar contents is common in blogosphere and this forms implicit links between bloggers. By measuring the text similarity of the blogposts, authors created a link graph between bloggers, and adopted the PageRank algorithm to rank the importance of bloggers.

Akritidis et al. [190] proposed a solution for bloggers identification in community blogsites that are both influential and productive. Based on the number of incoming links, two matrices are considered characterizing the bloggers as influential, productive, both, or none. This way, authors can identify blogger activities, temporal patterns and behavioral patterns. Eunyoung et al. [191] built a framework for identifying influential and popular bloggers by considering interpersonal similarity, which presents the interaction among bloggers and like-minded readers, and the degree of information propagation, which represents how many readers a blogger has. Their study showed that weighting blog social ties can differentiate influential bloggers from popular bloggers, and what make bloggers influential or popular. Finally, Cambria et al. [192] exploited the contents of social interactions to cluster OSNs based on semantics and sentics, that is the conceptual and affective information associated with the interactive behavior of OSN members. In particular, authors created a socio-interaction matrix encoding the concepts discussed by users and applied singular value decomposition (SVD) on it to perform user profiling and topic extraction.

3.2.2 Topic Extraction

Topic extraction is a technique used for discovering the abstract 'topics' that occur in a collection of documents, which is useful for tasks such as text auto-categorization, sentiment analysis but also SNA. Common approaches include mixture of unigrams, latent semantic indexing, LDA, and knowledge-drive methods [193]. Such methods can be used in combination with SNA to mine user interests in OSNs [194].

McCallum et al. [195] described how to determine roles and topics in a text-based OSN by building an author-recipient-topic (ART) model and a role-author-recipient-topic (RART) model. Pathak et al. [196] proposed a community-based topic model integrating SNA techniques termed community-author-recipient-topic (CART). This model was used to extract communities from an email corpus based on the topics covered by different members of the overall network. Later, Wang et al. [197] proposed an approach to predict the interests of new users or inactive users based on different social links between them using a random-walk based mutual reinforcement model that incorporates both text and links. Another work [198] proposed a regularization framework based on a relation bipartite graph, which can be constructed from any type of relationships, and evaluated it on OSNs that were built from retweeting relationships.

Kang et al. [199] presented a user modeling framework that maps user-generated content into the associated category of the news media platform. Similarly, Jipmo et al. [200] proposed a multilingual unsupervised system for the classification of Twitter users' interests. The system represents tweets and topical interests (e.g., technology, art, sports) as bags of articles and Wikipedia categories respectively, and orders the user interests by relevance, by computing the graph distance between the categories and the articles. Faralli et al. [201] proposed a method for modeling Twitter users using a hierarchical representation based on their interests. This was done by identifying topical friends (a friend represents an interest instead of social relationship) and by associating each of these users with a page on Wikipedia. Zarrinkalam et al. [202] proposed a graph-based link prediction system based on user explicit and implicit contributions to topics, relationships among users, and similarity between topics. Finally, Trikha et al. [203] studied the prediction of users' implicit interests based on topic matching using frequent pattern mining.

3.2.3 Sentiment Analysis

In recent years, sentiment analysis has become increasingly popular for processing social media data on online communities, blogs, wikis, microblogging platforms, and other online collaborative media [204]. While most works approach it as a simple categorization problem, sentiment analysis is actually a complex research problem that requires tackling many NLP tasks [205], including personality detection [206], domain adaptation [207, 208], and multitask learning [209]. It also comes in different flavors depending on granularity of the analysis [210], modality

adopted [211] and gender [212, 213]. Applications of sentiment analysis span domains like healthcare [214, 215], political forecasting [216], tourism [217], rumors and fake news detection [218, 219] and dialogue systems [220].

Sentiment analysis has also been applied to better understand OSN dynamics by looking at the exchange of information between network nodes. This can be useful for trend discovery, user profiling, influencer detection, and study of polarization over a certain topic or political orientation. One of the first works in this context was [221], which presented an approach for analyzing knowledge perception in an OSN. Authors provided an evidence-theory based methodology for constructing and maintaining a knowledge network in an electronic-mail communication environment. Later, [222] focused on the topic of online radicalization. Authors monitored users and interactions in a specific YouTube group using a combination of sentiment, lexical and SNA techniques. They made a number of interesting observations about the differences in the nature of the discussion and interactions between male and female members of the group.

[223] studied the blogosphere's sentiment towards Barack Obama during the 2008 USA presidential elections. Authors used a hybrid machine learning and logic-based framework. The results showed that the classification task in this environment is inherently complex, and learning features that exploit entity-level sentiment and social network structure can enhance classification. [224] proposed an ensemble of sentiment analysis and SNA for extracting classification rules for each customer. These rules represent customer preferences for each cluster of products and can be seen as a user model. This combination helped the system to classify products based on customer interests. Authors compared the results of their method with a baseline method with no SNA. Experiments on an Amazon meta-data collection showed improvements in the performance of the classification rules compared to the baseline method. More recently, [225] proposed an ensemble of sentiment analysis and SNA tools for group decision making. In particular, sentiment analysis was used to model consensus among experts in an OSN and later exploited to automatically generate preference relations, which were used for carrying out the group decision making process.

4 Application domains

Due to the number of application domains in OSNs is extremely large, we have selected a subset of them by the scientometric study performed in Section 2. This study allowed us to select the following application domains: Healthcare (Section 4.1), Marketing (Section 4.2), Tourism and Hospitality (Section 4.3) and Cyber Security (Section 4.4). Also, we have identified two emerging areas of SNA that are described in Section 4.5: Politics and Detection of fake news and misinformation.

4.1 Healthcare

Over the last years, there has been a huge growth of research papers that study the applicability of OSN on health [226, 227, 228]. It is widely recognized that social relationships have powerful effects on physical and mental health [229, 230, 231]. With the integration of OSNs into our daily life, new levels of social interaction have emerged, and new possibilities beyond the traditional doctor-to-patient paradigm have arisen. Many patients with different diseases are now using OSNs to share experiences with other patients with similar conditions, providing a new potential source for acquiring knowledge very useful [232]. In addition, there are numerous health-related behaviors that might easily spread in OSNs, such as obesity, smoking, alcohol consumption, or drug use. Therefore several research studies have been appeared analyzing this effect from the SNA perspective [226].

The propagation and virality models in OSN have been one of the most used techniques for analyzing health-related behaviors in OSNs. For example, a study carried out by Christakis & Fowler [233] used a global network design to analyze how **obesity** can spread through OSNs in a similar way to a infectious disease. For this purpose, the authors evaluated a densely interconnected OSN of $12,067$ people who underwent repeated measurements over a period of 32 years (from 1971 to 2003) as part of the Framingham Heart Study. Then, a quantitative analysis of the nature and extent of the person-to-person spread of obesity as a possible factor contributing to the obesity epidemic was performed. This study examined several aspects of the spread of obesity using OSN, including the existence of clusters of obese persons within the network, the association between one person's weight and height gain among his or her social contacts, the dependence of this association on the nature of the social ties (e.g., ties between friends of different kinds, siblings, spouses, and neighbors), and the influence of sex, smoking behavior, and geographic distance between the domiciles of persons in the OSN. The results shown in this study suggests that obesity may spread in OSNs in a quantifiable pattern that depends on the nature of social ties.

Smoking and alcohol consumption among adolescents are prominent risk behaviors for Health in the United States [234], and there is also substantial evidence that adolescents' use of tobacco and alcohol is highly associated with their friends' use [235]. OSNs provide a mechanism for adolescents to connect with friends instantaneously, and several research work has been focused on toward uncovering the risks associated with the usage of OSNs, such as the display of

inappropriate content like sexual references and substance use [236] or tobacco advertisements [237]. Following this research line, Huang et al. [238] carried out a study to investigate peer offline and online friendships, for determining how online activities with friends might broker the peer influence processes, by either encouraging or hindering the influence of peer risk behaviors on adolescent smoking and alcohol use. For this purpose, a friendship network data about adolescent social media use and risk behaviors, were collected from $1,563$ 10th-grade students across five Southern California high schools, and measures of online and offline peer influences were calculated and assessed using linear regression models [239], which were fitted to test the effects of online activity with friends on smoking and alcohol use indicators. The results of the analysis showed, that the frequency of adolescent using OSNs and the number of their closest friends on the same OSN were not significantly associated with risk behaviors. However, exposure to online pictures of their friends of partying or drinking, was significantly associated with both smoking and alcohol use. Whereas adolescents with drinking friends had higher risk levels for drinking, adolescents without drinking friends were more likely to be affected by higher exposure to risky online pictures. Other study on the extent of the person-to-person spread of smoking behavior was introduced by Christakis and Fowler [240]. The authors analyzed a densely interconnected OSN of $12,067$ people assessed repeatedly from 1971 to 2003 as part of the Framingham Heart Study, applying network analytic methods and longitudinal statistical models to detect clusters of smokers and nonsmokers in the network. The findings of this analysis show that smoking behavior spreads through close and distant social ties, groups of interconnected people stop smoking in concert, and smokers are increasingly marginalized socially.

There are other works that use data from the OSNs to extract valuable information. For example, the text messages posted in OSNs offer new opportunities for the real-time analysis of expressed mood and social behavior patterns [241]. The analysis of these additional types of information can be very useful to the early identification, assessment, and verification of potential public health risks [242], and the timely dissemination of the appropriate alerts (public health surveillance) [243]. For instance, Twitter users may post about an illness, and their relationships in the network can give us information about whom they could be in contact with. Furthermore, user posts retrieved from the public Twitter API can come with GPS-based location tags, which can be used to locate the potential centre of disease outbreaks. Therefore, a high number of research works have already appeared, showing the potential of Twitter messages to track and predict outbreaks. Based on this idea, a document classifier to identify relevant messages was presented in Culotta et al. [244]. In this work, Twitter messages related to the flu were gathered, and then a number of classification systems based on different regression models to correlate these messages with CDC statistics were compared; the study found that the best model had a correlation of 0.78 (simple model regression). Other similar approach was proposed by Aramaki et al. [245] where a comparative study of various machine-learning methods to classify tweets related to influenza into two categories (positive and negative) was carried out. Their experimental results showed that the support vector machine (SVM) model that used polynomial kernels achieved the highest accuracy (F-Measure of 0.756) and the lowest training time. Bello et al. [61] focused the research on the detection and tracking of discussion communities on vaccination arising from OSNs as Twitter. Finally, well-known *regression models* were evaluated on their ability to assess disease outbreaks from tweets in Bodnar et al. [246]. Regression methods such as linear, multivariable, and SVM were applied to the raw count of tweets that contained at least one of the keywords related to a specific disease, in this case “flu". The models also validated that even using irrelevant tweets and randomly generated datasets, regression methods were able to assess disease levels comparatively well.

4.2 Marketing

In social media, brands and customers co-exist in such a way that both can interact with each other. On the one hand, companies can use social media to promote new products or to predict what the customers think about the brand or the product. On the other hand, customer can express their opinions and doubts with other customers. For all of that, social media has become the favorite and most popular platform to perform promotional activities to communicate with the targeted customers [247, 248].

There is a large number of works that have studied the different aspects related to promotional activities conducted in social media platforms [249, 250, 251, 252]. For example, it has been studied the relation between how customers perceive and formulate their attitudes towards social media advertising activities, and how these perceptions affects to the efficiency and effectiveness of such activities [253]. Other work [254], stated that if brands want to generate positive customer attitudes, they have to carefully address hedonic aspects to provide customer pleasure experiences.

What is clear is the contribution of social media in the generation of the new Word-of-Mouth, *e-WOM* that stands for *electronic word of mouth*. Using social media, customers are able to express their opinions about specific products or brands to many other customers [255, 256, 257, 258]. One of the most famous works regarding e-WOM is the one performed by Jansen et al [259]. This work studied whether Twitter could become used as an e-WOM advertising mechanism. The conclusion of this work, revealed that 19% of the tweets analyzed mentions a brand, and 20% of these tweets contained expressions of brand sentiments. The conclusions of this work are that (1) Twitter reports what

customers feel about the brand and its competitors, and (2) customers' brand perceptions and purchasing decisions are influenced by social media services. Other study is the one published by Asur and Huberman [260]. In this case authors used Twitter to forecast box-office revenues for movies. In order to do that, authors built a model based on the rate at which the tweets were created. The results reveals that this model outperforms the market-based predictors.

The rate at which the tweets, or post, are created also foster the level of interactivity and association with their customers [261, 262] and thus, firms also use social media to contribute to both customer experience and customer relationship management [263]. In this sense, several authors have studied the usage of social media for the just mentioned purpose [264, 265, 255, 156] and the general conclusion is that social media is a good tool to help organizations to sustain their relationship with their targeted customers. Nevertheless, research community has also identified that not all OSNs contribute in the same way. Moore et al. [266] concluded that this role of social media in forming customers' relationship with brands could be different according to the kind of platform used: Facebook or Twitter. In this sense, Pereira et al. [267] noticed that though customers on Facebook are enthused to follow brand accounts, they are less interested in keeping contact with them or re-sharing their content in their own page. Similar research were conducted on Twitter by Kim et al. [257], where based on the data collected by the authors, the conclusion is that brand re-tweeters show an extent level of brand trust, brand identification and community commitment.

4.3 Tourism & Hospitality

Tourism and Hospitality is other area that can take advantage of the usage of social media to extract valuable information [268, 269]. On the one hand, hotels and tourism destination use social media with Marketing purposes [270, 271]. On the other hand, it has been probed that half of tourists change their travel plans after studying their trip on social media [272].

Tourism research has paid attention to user generated contents on social media to extract some valuable information. In this sense, there area two different data sources taken into account: (1) online textual data, published by customers on social media, and (2) online photo data. The research procedure will depend on the different source of data analyzed (text, images or both), which directly will affects to the algorithms that can be applied, and to the knowledge and patterns that can be extracted [273, 274, 275].

As in the Marketing area, tourists use OSNs to express their opinions about the different places, their experiences, and their satisfaction and dissatisfaction about tourism products. Analyzing these data, researchers have discovered relevant attributes of tourist satisfaction [276, 277], the relations between tourist satisfaction and other related factors, such as guest experience and competitive position [278], or how the users use the OSNs for evaluating and improving the e-WOM hotels [279, 278]. Nevertheless, the majority of the research works published are focused on investigating the customers opinion and customers experience. In this sense, Guo et al. [277] used LDA to discover the aspects that influence the customers satisfaction. Similarly, Poria et al. [280] proposed Sentic LDA, an affective version of LDA based on SenticNet [281], which leverages the semantics associated with words and multi-word expressions to extract the polarity associated to tourism-specific aspects such as accommodation, entertainment, food, and transportation [282]. Other authors used sentiment analysis to extract tourists attitude and opinion toward tourism products such as hotel services [283]. The work published by Bordona et al. [284] used clustering to group together those trips with similar `geoslotID` based on the geo-tagged messages in Twitter.

The second data type, which has been used to analyze tourism in social media, is the photos published by tourists. In this sense, the most extended approach is to analyze the metadata associated with the photo, instead of working with the photo itself. Using, for example, the geo-tag information researchers have been able to explore the tourists behavior in Hon Kong [285], to create a recommendation system that provides travel paths [286], or to select photo elements from the viewers' perspective and assist marketing organizations [287]. There are other metadata associated with the photos, for example, the user id and the photo id, the date and time when the photo was taken and uploaded, the geographical information expressed by the latitude and longitude, and other information such as the title, descriptions or tags of the photo. On the one hand, there is a wide variety of papers that uses these metadata and clustering techniques to group the different photos analyzed. In this domain it is quite popular to use a density-based spatial clustering to solve the problems related to centroid-based approaches [283, 288]. On the other hand, other researchers focus their work on travel trajectories, i.e., the sequence of tourism spots and time intervals between them to recommend travel plans for tourists. For example, [286] suggests different routes taking into account the quality and popularity of the routes. Other approach is the one followed by Vu et al. in [285], where authors applied Markov models to predict the next tourism spots based on the current location of the tourists.

4.4 Cyber Security

SNA can be used to detect and apply different strategies to support law enforcement agencies in the fight against cyber-crime and cyber-terrorism. Criminals, and terrorists, use OSNs, such as Twitter or Facebook, due to the huge number of users that connects everyday to these networks. Moreover, the internal structure of these networks, make that any published message propagates very fast, reaching a high number of potential readers. In this domain, the purpose of SNA techniques and algorithms is to extract the different patterns of criminals and terrorist in OSNs. More precisely, the main goal is to detect and discover crimes and their relations with criminals.

Regarding crime analysis, huge efforts have been made to facilitate the communication between citizens and government agencies. Initially, this communication was roughly performed through telephones, or face-to-face meetings. Then, the information provided by the citizens was saved, or transformed into written text and then archived in a digital format. In order to automate and facilitate crime analysis, Chih-Hao and Gondy [289] designed a decision support system that combines NLP techniques, similarity measures, and classification approaches. Filtering reports and identifying those that are related to the same or similar crimes can provide useful information to analyze crime trends, which allows for apprehending suspects and improving crime prevention.

Other works have been focused on the prediction of the different crimes by trying to discover hidden patterns of criminal behavior. In this sense, it is quite popular geographic knowledge discovery techniques that can be used to discover these patterns that may help in detecting where, when, and why particular crimes are likely to happen. In order to do that, Phillips et al. [290] presented a crime data analysis technique that allows for discovering co-distribution patterns between large, aggregated, heterogeneous datasets. Authors modeled the aggregated dataset as a graph that stores the geospatial distribution of crimes within given regions. Then, these graphs were used to discover those regions with similar geospatial distribution characteristics.

A different approach is the one followed by Chainey et al. [291]. This approach is based on the visual identification of regions where the crimes can be produced by using the *hotspot mapping*. This technique is used to predict where crime may happen, using data from the past to inform future actions. Each crime event is represented as a point, allowing for the geographic distribution analysis of these points. A number of mapping techniques can be used to identify crime hotspots, such as: point mapping, thematic mapping of geographic areas, spatial ellipses, grid thematic mapping, and kernel density estimation (KDE), among others. In the just mentioned work, authors performed a comparative study of these mapping techniques and the results revealed that KDE was the most successful technique. This conclusion was also extracted from the work performed by Gerber [292], where KDE was used to automatically identify discussion topics across a city in the United States by applying linguistic analysis and statistical topic modeling over a spatio-temporally tagged tweets extracted from Twitter.

Different approaches are the ones applied to analyze the terrorists' behavior in OSNs. From the last years, it is quite simple to realize the ability of terrorist groups in the usage of OSNs for their own benefit [293, 294]. Terrorist organizations use OSNs to promote their ideology, and to recruit individuals to their cause. Usually, the first conversations start in the most famous OSNs such as Twitter, Facebook, or Instagram, and then they continue using private message with the target individuals. In order to disconnect these radicalization channels, governments, organizations and social media platforms are continuously searching social media accounts that can be associated with such terrorist groups to block them.

From the research community, a wide range of works has been done in order to help law enforcement agencies. In this paper we are going to talk the four main application domains related to counter-terrorism in OSNs. The first one is focused on the identification and understanding of the language used, or the definition of the linguistic markers. The key idea is the analysis of the text published in the OSNs in order to determine whether the corresponding account belongs to a terrorist or, at least, a supporter. Cohen et al. [295] discussed about the possibility of detecting some of the linguistic markers defined by Meloy et al. in [296]. More precisely, Cohen et al. discussed about the detection of *leakage*, *fixation* and *identification* warning behaviors because these are the ones have the greatest potential to be discovered with text analysis methods. In a more practical way, Torregrosa et al. [297] compared the tweets published by pro-ISIS Twitter accounts against the text published by random Twitter users. To analyze the terminology of the tweets they have used Linguistic Inquiry Word Count (LIWC) software. Experimental results reveals that pro-ISIS accounts publish tweets using the third person plural pronouns, they use more words related with death, certainty, and anger, and more negative language than random accounts. Some differences in the language was also evinced by Lara-Cabrera et al. [63]. They also compared the tweets published by pro-ISIS accounts and random Twitter accounts, using a set of keywords that was extended with different synonyms, and taking into account the stem of the words. Experimental results revealed that metrics defined to measure the indicators performs well in the tasks of identifying those accounts that uses a radical vocabulary. Once the user has been identified as terrorist, or supporter, it is possible to study how this user influence the other users in the OSN [298].

All these works, help to highlight those social media accounts that shows a radical behavior based on the content of the tweets. Other approaches model the OSN in a graph and use connections to detect the critical node [299]. For example, Gunasekara et al. [300] uses the betweenness centrality of the nodes to detect the critical nodes of the graph. The main problem with the betweenness centrality metric is that the metric is really expensive from the computational point of view, it runs in $O(nm)$. In order to alleviate this problem, other works uses heuristic or bio-inspired approaches such as Lozano et al. [301] that integrates an updated procedure of betweenness centrality metric with an artificial bee colony algorithm.

4.5 Emerging Areas

Finally, and out of the scope of our scientometric analysis carried out in Section 2, we have decided to make a brief analysis of the state of the art in some emerging areas, that are currently experiencing an increasing interest in the area of OSNs. These emerging areas are directly related to recent, but highly societal demanding topics, such as politics and detection of fake news and misinformation, or the integration of multimedia information. With the irruption of 5G and IoT technologies [302, 303], it would be expected that in the next years the concept or social Internet of Thinks [304] will generate a huge interest in the field of SNA allowing to generate new kind of intelligent services to end-users through the combination of Machine Learning, Artificial Intelligence and IoT methods. We strongly think that these areas could be in a near future high activity niches, for the research communities involved in areas as SNA, Data Science or Big Data.

4.5.1 Politics

The Arab Spring in 2011, and both Obama's campaigns (in 2008 and 2012), marked the beginning of how social media might affect citizens' participation in political life [305]. Since these dates, politicians, citizens and researchers have expressed their interest in how they can take advantage of using social media to participate in political life. In this sense, politicians use social media to attract supporters, and people have been using it to express their political views and opinions about various leaders and issues. In fact, Boulianne [306] studied whether social media use, and participation in elections, were correlated. Paying attention to the metadata, more than 80% of the coefficients showed a positive relation.

The research community has used social media to study a wide variety of problems. The most remarkable are the following: (1) hate speech detection [307, 308]; (2) topic opinion, or political polarization [162, 309]; (3) community finding problems [17, 310]; and (4) information exchange and information diffusion [12, 311]. The majority of the research works start with the same idea of analyzing the text of the posts, or comments, and using the results of this analysis to perform a second analysis more focused on the problem to be solved.

Hate speech is used in those posts or comments that defames, belittles, or dehumanizes a class of people on the basis of certain inherent properties such as race, ethnicity, gender, or religion. Due to its goal, it is critical to design systems that automatically classify the posts or comments based on its content and determine whether the corresponding post contains hate speech or not [307, 308]. In Müller and Schwarz demonstrated that there is a significant correlation between increased German hate speech on social media, and physical violence towards refugees in Germany [312]. One recent work that tries to identify hate speech is the one published by Jaki and De Smedt [313], where authors tries to understand what disparaging verbal behavior from extremist right wing users looks like, who is targeted and how. In order to do that, they analyzed 55.000 right-wing German hate tweets from August 2017 to April 2018. The proposed method is able to detect right-wing hate speech with 84% accuracy. Other relevant work in this domain is [314], where authors used several classifiers to detect hateful and antagonist content in Twitter. Although the individual results for each classifier reduce false positives and produced promising results regarding false negatives, the combination of classifiers into an ensemble classification approach seems to be the most suitable method.

There are other works that are focused on the analysis of the content of the tweets, or posts. In this case, we can talk about topic opinion or political polarization. Although these concepts have different goals, the procedure followed is quite similar. In both cases, works try to extract some valuable information from the content of the tweets. On the one hand, if the goal is to understand what the citizens think about specific topics, we talk about *topic opinion*. On the other hand, the goal of *political polarization* works is to discover the alignment of the citizens with the corresponding parties. In this sense, [315] belongs to topic opinion family. Authors analyzed $1,150,000$ messages from about $220,000$ users to define the characteristics of the three main parties in the 2010 UK General Election and highlight the main differences between parties. In a first analysis they realized that: (1) the retweet structure is highly clustered according to political parties, (2) users are more likely to refer to their preferred party and use more positive affect words for the party compared with other parties, and (3) the self-description of the users can reflect the political orientation of users. Based on these evidences, authors developed a classification method that uses the number of tweets referring to a particular political party, and its semantic content to estimate the overall political leaning of the user. The experimental results

achieved an accuracy of 86% for classifying the users' political leanings. Other study [316] tries to determine whether the Twitter users can be grouped around the different parties during the elections. To do that, they analyzed around $6,000$ tweets published by $1,500$ Twitter users during the 2011 Canadian Federal Election. Authors concluded that Twitter usage is likely to further embed partisan loyalties during electoral periods rather than loosen them. Therefore, it seems that partisan are closer to their corresponding parties during electoral periods. Other works try to predict the vote intention of the citizens based on their public posts. This is the case of [317], where they tried to classify people's voting intentions based on the content of their tweets during the Scottish Independence Referendum in 2014. They built a topic-based naïve bayesian model (TBNBM) that takes into account the dependencies between topics and user voting intentions. This TBNBM detects the topics using the LDA, and for each topic they built a probability table where each feature has two associated conditional probabilities related to both voting intentions (i.e., 'Yes' or 'No'). In the experimental phased authors realized that this TBNBM improves the classical bayesian classifications. Finally, other relevant work is the one published by Borge-Holthoefer et al. [318]. In this case, authors used social structured and the content of the tweets to understand the opinion evolution in Egypt during the summer of 2013. They observed that the military takeover caused major quantitative (volume of polarized tweets), but not ideological (polarity swaps) shifts among Twitter users. They also observed how the pro-military Twitter users that were very loud before the take-over, became increasingly silent afterwards, and how anti-military intervention Twitter users become significantly louder after the takeover.

Some of the just mentioned works can be also categorized into community finding problems because they detect some clusters, or communities, of users based on different aspects. In this domain, it is important to highlight the work published by Ozer et al. [319] because they developed three Non-negative Matrix Factorization frameworks to investigate the contributions of different types of user connectivity and content information in community detection. They revealed that user content and endorsement filtered connectivity information are complementary to each other in clustering politically users into pure political communities. Other work [320], predicted the political alignment of Twitter users based on the content and structure of their political posts after the 2010 U.S. midterm elections. They used manually annotated data and the different communities are created by taking into account the TF-IDF and the hastags obtained by content analysis. They found a highly segregated partisan structure with few retweets between left and right-wing Twitter users.

Finally, some other works are focused on how the information is propagated through the network. In this regard, a well-known work is the one published by Colleoni et al. [321]. In this work, authors used a combination of machine learning and SNA to classify users as Democrats or Republicans based on the content shared on social media. Then, they investigated the political homophily in both: the network of reciprocated and non-reciprocated ties. They found that the structures of political homophily differ significantly between Democrats and Republicans. Other works try to study whether the sentiment of a tweet affects to its propagation [322], i.e., how the affective dimensions of tweets, including positive and negative emotions associated with certain political parties or politicians, affect the quantity of retweets. Experimental results over around 64.431 political tweets reveals a positive relationship between the quantity of words indicating affective dimensions (positive and negative) and its retweet rate. The last remarkable work [323], authors studied whether the online communication of political and nonpolitical issues resembles an "*echo chamber*" (i.e., communication between individuals with same ideological segregation) or a "national conversation". In order to do that, authors analyzed around 150 million tweets regarding 12 political and nonpolitical issues, extracted from 3.8 million Twitter users. The findings suggest that in terms of political issues, the information was exchanged primarily among individuals with similar ideological preferences; but this effect did not happen with non-political issues.

4.5.2 Detection of Fake News & Misinformation

In 2018, 66% of American adults consume news on social media[2], whereas in 2012 only 49% of adults used OSNs for this purpose. This increment is due to two main reasons. The first one is related to the fast propagation of messages on OSNs. The second reason is related to the high number of users connected to any OSN which makes quite easy to comment and discuss any message. Due to the easiness of creating messages and disseminating them in any OSN, it has been significantly increased the number of *fake news* [167, 324, 325], i.e., those news with intentionally false information produced online for a wide range of purposes, such as financial and political gain [166, 326].

In spite of its popularity, the research community is not able to agree in a common,and unique, definition of the term *fake news*. The most extended definition is the one that states that fake news are those news articles that are intentionally false, its truthfulness can be verified, and its intention is to mislead readers [166, 327, 328]. This definition is based on two key concepts: *authenticity* and *intent*. The former means that fake news contain false information that can be verified. And the latter refers to the fact that fake news are created to confuse and mislead customers.

[2] `http://journalism.org/2018/09/10/news-use-across-social-media-platforms-2018`

The main risk about fake news is related to the concept of *intent*, just described. The goal of any fake news is to persuade consumers to accept biased, or false, information usually with political messages or influence. This goal and the fast propagation of messages in OSNs made that, for example, the most popular fake news was even more widely spread on Facebook than the most popular authentic mainstream news during the U.S. 2016 president election[3]. For all of that, it is really important not only to understand how fake news propagates through the network (see Section 3.1.3) but also to develop systems that detect whether a specific new is a fake news or a real one [167].

Fake news detection is a promising area that tries to define whether a specific new is fake or not [167, 327]. There are different approaches that can be followed to do that, but all of them can be grouped in two big categories: the *linguistic approaches* and the *network approaches*.

The goal of *linguistic approaches* is to detect the fake news by analyzing its context. The idea is to obtain the different writing styles to detect fake news. In this sense, research works focus on two different linguistic features: (1) *lexical features*, such as total words, characters per word, frequency of large words, etc. [329], and (2) *syntactic features*, such as "n-grams" and bag-of-words (BOW) or parts-of-speech (POS) tagging [330].

Second category, *network-based approaches*, are based on the network that can be built by taking into account the users that publish related social media posts. In this case, researchers create a network based on some specific interaction on the OSN, and then, they apply network metrics to extract the valuable information. There are different types of network that can be built. For example, the *stance network* is a graph where the nodes represent all the tweets relevant to the news, and the edges contains a weight that indicates the similarity of stances [331, 332]. A different approach is the one followed in the *co-occurrence network* where two nodes are connected by a weighted edge that represent how many times both users have written post relevant to the same news articles [333]. And the last example is *friendship network* users who posted related tweets are connected in the network[334].

Finally, once the model is built, the network metrics can be used to extract valuable information. For example, authors of [334] used degree and clustering coefficient to characterize the diffusion network. A different approach is the one followed in [333] where SVD is used to learn the latent node embedding.

4.5.3 Multimedia

Nowadays, although users are able to generate different types of content in OSN like audio, image or video, text is the most common, and popular, content analyzed. The analysis of audio, image or video, has received less attention compared to text mainly due to the complexity of the analysis tasks, and the initial technological limitations of internet in its early years. The internet connections were slow and encoding algorithms were poor, thus, sharing videos, audios and images were impractical. Nowadays, internet connections have improved and smartphones, all equipped with cameras and microphones, have emerged as a portable alternative to traditional computers with access to the internet. As a result, it is very easy to share videos, photos, or audio using those devices, in fact, some of the most popular modern OSN are completely based on sharing images and videos like Instagram or Pinterest. Consequently, the interest in analyzing multimedia content in OSN has suffered an important increase.

Taking into account the different multimedia types audio, video and images, it is the latter the most common data type analyzed. Some of the works found in the literature have similar goals as the ones analyzing texts, such as Sentiment analysis [335]. Furthermore, Image Annotation [336], which is generating a set of words that describe the content of a picture, is also commonly applied and it has similar goals to the keyword extraction methods. Besides, Image Clustering [337] can also be found, and it would be the equivalent to topic extraction. Although the just mentioned approaches share the same goals as some of the most popular text-based algorithms, the methods based on images are more complex due to the complexity of processing images. One characteristic of the images, that it is not present in the text analysis, is the fact that images contain information about the geographical location where the image was taken. This fact has allowed SNA researchers to use images as geo-location alternatives.

There is a plethora of works that analyze images for extracting some valuable information. For example, it is possible to predict the popularity of a picture [338, 339], to predict the gender of users [340], to discover events in public places [341], or forecast the ambiance of a place [342], by applying Sentiment analysis and Image Annotation. Other works, like [343], have used Image Clustering to analyze the content of popular images in OSN, such as pets, food, ... etc. There is other wide set of works that have developed their own algorithms for analyzing faces and predicting human characteristics [344, 345] or perceived intelligence [346]. Finally, other works need the supervision of humans to validate the output of the algorithms. Some examples of these works are the one that analyze marijuana-related content [347] or cyber-bullying content [348].

[3] `http://buzzfeednews.com/article/craigsilverman/viral-fake-election-news-outperformed-real-news-on-facebook`

It is possible to classify the different algorithms and methods developed to perform some image analysis in three different groups: (1) Crowdsourcing, (2) Deep Learning, and (3) Handmade Features. On the one hand, Crowdsourcing [349] consists of the division of a work package into small pieces, or sets of images, and distributing them between a large number of participants, humans, to achieve cumulative results fast. Usually, this task is performed using some online platform that acts as an intermediary between the owner of the dataset and the workers. Popular platforms for doing image analysis based on Crowdsourcing are Amazon Mechanical Turk[4] (MTurk), Figure Eight[5] (formerly CrowdFlower) or MicroWorkers[6]. On the other hand, Deep Learning and Handmade Features avoid human intervention and use algorithmic approaches to extract significant features from images to be used as representatives. Based on this idea, Handmade Features use custom human-made filters that allow characterizing an image by low-level features, such as, colors [350, 351], shapes [352, 353], and textures [354, 355], or by high-level ones like the Semantics-Preserving Bag-of- Words (SPBoW) model [356] or the Contextual Bag-of-Words (CBoW) one [357]. Deep Learning, specifically convolutional neural networks (CNNs) [358], automatically construct these filters on their own. To this end, a supervised learning approach is used to generate a series of convolutions, ordered as layers connected one to another, capable of compressing an image into a representative set of characteristics. It is common to find works that use deep CNN (those with several layers), already trained to identify a huge number of different types of pictures (elephants, cars, plants, houses ... etc). For example, in [359] authors trained CNN with the ImageNet dataset. Finally, once the characteristics are extracted, common supervised/unsupervised machine learning techniques like neural networks [360], SVM [361], k-means [362] or DBSCAN [363] are used for doing particular analyzes.

Audio analysis in OSN has been largely untapped so far. It is quite common to apply Automatic Speech Recognition [364] to transcript the audio into a text and then, applying Speech Sentiment analysis extract some useful knowledge [365]. In spite of, for the best of the authors' knowledge, there are no works that have used these techniques to SNA, there are some applications of audio analysis in OSN. The first one is related to music. The Internet has been influencing the music scene for the last few decades, making music more accessible to the public and engaging musicians with their audience more easily. Thus, several social media platforms exist focused on music like Last.fm[7] or SoundCloud[8]. With this in mind, [366] proposed to use audio analysis to categorize music into genres to enhance community finding techniques. The other main application of audio analysis is generating conversational networks from raw audio data [367, 368]. Once the conversational networks are generated, SNA techniques are used to perform speaker role recognition [369, 370] and summarizing [371] on broadcast news and podcasts.

Video analysis is, by far, the media type least studied in the context of OSN. It is also the most complex media type to analyze from the three mentioned in this section (image, audio, and video) because it encompasses the other two. However, video media content is prevalent on the Internet and the most popular OSN are completely centered on it, like Youtube or Youku (YouTube's Chinese counterpart). Although, it can be found several papers focused on the video analysis, these techniques have not been applied yet in conjunction with SNA techniques. One example is the work published in [372] where authors apply sentiment-analysis to the video once it had been transcribed.

5 Discussion on Research Methods and Application Domains

From the analysis of the state of the art in both research and application areas of SNA, shown in Sections 3.1 (OSN Structural-based analysis: graph theory, community detection algorithms, information diffusion), 3.2 (OSNs Content-based analysis: topic extraction, opinion mining and sentiment analysis, multimedia), and Sections 4.1 to 4.5 (health, marketing, tourism and hospitality, cyber security, politics, fake news and misinformation, 5G and IoT technologies), some important conclusions can be drawn:

- On the one hand, OSNs are used as the highest socially trending, and influencing, source of information. Currently these sources have become on the most popular ones in the world, with billions of active users generating huge amount of data (in form of textual information, video, and other multimedia material) per second.
- Working with only a small part of the available data stored in any OSN, exceed the classical processing algorithms capabilities, and suppose one the biggest current challenges for a wide number of research areas. The complexity of this domain needs by a joint effort from different research areas, so the *multidisciplinarity* between areas, from Social Sciences to Science and Engineering, becomes a necessity to find new methods and technologies to exploit the enormous potential of these sources.

[4] `http://mturk.com`
[5] `http://figure-eight.com`
[6] `http://microworkers.com`
[7] `http://last.fm`
[8] `http://soundcloud.com`

- Although multidisciplinarity is a necessity, it means that professionals from very different areas, with very different background, have to collaborate together on a wide range of technologies (e.g., sociologists and data scientists).
- On the other hand, most of algorithms and methods studied attempt to address the set of **challenges** (gathering, and processing the data, finding useful patterns, visualize the information, etc.) imposed by the complexity of this domain, in particular:
    1. Both structural-based and content-based algorithms, are used to *extract or discover* useful knowledge from the network. This challenge is related to the classical Knowledge Discovery problem in Machine Learning, Data mining and Data Science fields. This challenge, or problem, is directly related to our proposed RQ1 (*What can I learn?*).
    2. The necessity to manage a huge, and exponentially growing amount of data, needs from new methods, *scalable* algorithms, and technologies. This is one of the hot topics in areas as Big Data, and it is related to RQ2 (*What is the limit?*).
    3. Due to the vast number of sources available, and their different data formats (numerical, categorical, textual, metadata, video, images, audio), it's necessary to develop new algorithms capable to *integrate* and *fusion* different sources to allow discovery new and useful knowledge. This problem is usually address by the area of Information fusion, and it's related to RQ3 (*What kind of data can I integrate?*).
    4. Finally, one of the essential tools for knowledge discovery in OSNs is related to the *visualization* of the information. This is a challenging, and still open problem, in the area of Information Visualization, so the RQ4 (*What can I show?*) has been defined to cover this aspect.

Taking into account previous conclusions, this work proposes the definition of four **dimensions**, which can be used to assess the maturity of technologies currently available in OSN. These four dimensions will later be used to define a set of **metrics** (which we named $degrees_*$) that will be used:

- *To quantitative assess the level of maturity of a set of SNA tools and frameworks*. To do that, a set of graphical representations (based on spider graphs), and a new global metric (named $\mathfrak{C}_{SNA}$), will be defined to provide a quantitative measure of these tools and frameworks.
- *To study some possible future trends, challenges, and lines of work in these dimensions* (closely related to research areas such as Data Science, Big Data, Information fusion and Visualization), detecting spaces for improvement, and emerging technologies, where new developments could have a high impact on the scientific community, industry and society.

6 The Four Dimensions of Social Network Analysis

The Big Data paradigm was characterized by the different V-models that allow any researcher to analyze the capacity of the different Big Data methods. Initially, 3Vs were described in the 3V model [14], but this model has evolved during the last years to the 4V [373, 374], 5V [375], or 6V model [376]. These models allow measuring the maturity of different methods, tools and technologies based on Big Data using simple features such as Volume, Velocity, Variety, Value, Veracity or Variability. There are even some attempts to include new 'V's likes Visualization, in these *V-models. This set of *V-models provides a straightforward and widely accepted definition related to what is (and what is not) a big-data-based problem, application, framework, or technology. We will use this interesting (and successful) approach to describe the challenges, and the current status, of technologies related to OSN. We have mapped our four Research Questions stated in Section 1, into a set of equivalent dimensions: D1) *Pattern & Knowledge discovery*, D2) *Scalability*, D3) *Information Fusion & Integration*, and D4) *Visualization*. Using these dimensions, we can quantify the different methods, techniques, algorithms and frameworks, allowing us to better understand where we stand and where we could be in the near future in this area.

6.1 Pattern & Knowledge Discovery (D_1)

The concept of *Value*, as the knowledge or pattern discovery capacity of any method or algorithm, can be easily extrapolated to the area of SNA. This first dimension will be used to define the capacity of Knowledge discovery (mainly from a pattern mining perspective) of SNA technologies. This dimension tries to answer the question: *What can I learn?*, understood as the capacity to discover non-trivial knowledge from OSN.

Knowledge discovery and pattern mining is one of the central topics in different areas as Data Mining [377, 378], Big Data [7, 379], and Data Science [380, 381], which can be considered as a new paradigm that includes everything from

Big Data Analytics, Data Mining, Predictive Modeling, Data Visualization, Mathematics, to Statistics. In the area of Big Data, this topic is related to the concept of Value and usually refers to the process of extracting valuable information from very large data sets (e.g., from social big data [17]), and it is usually referred to as Big Data Analytics [382, 383, 384].

Any Data Mining process can generate a large number of patterns from data (from some few ones to hundred or thousands), therefore in Data Mining it is necessary to select or filter those useful patterns from the trivial or redundant. In such case, it is possible to use a set of measures designed for evaluating and ranking the discovered patterns produced by the data mining process. In Data Mining area, this set of measures are usually named as *interestingness measures* [385, 386, 387], which are intended for selecting and ranking patterns according to their potential interest to the user. This kind of measures are used to reduce the number of rules discovered by users, so they are used to restrict the final number of rules that are, or could be, interesting to the user. Although there is wide number of interestingness concepts that has been defined, most of them emphasizes conciseness, coverage, reliability, peculiarity, diversity, novelty, surprisingness, utility, and actionability [385]. These interestingness measures can be divided into three main categories [388]: (1) *objective measures* [389], which are based only on the raw data, and the statistical strengths, or properties, of the discovered patterns; (2) *subjective measures* [387, 390], which take into account both the data and the user of these data, and are derived from the user's beliefs or expectations of their particular problem domain (so they are based on the subjectivity of the user who evaluates the patterns on the basis of novelty, actionability, unexpectedness, etc.); (3) and, *semantic-based measures* that considers the semantics and explanations of the patterns. The main problem of all these measures is that they cannot be applied directly to quantify the dimension relative to the value provided to the discovery of knowledge by the different techniques and tools used in SNA. These metrics allow us to quantify the value of the resulting patterns that are generated by a specific data mining process after being applied. In other words, a specific technique must be applied to a specific set of data, and the patterns obtained as a result are evaluated. In addition, for each type of mining technique, there are different measures to evaluate its patterns. Therefore, these measures will not able to be directly applied to quantify the value in SNA techniques and tools.

The objective of this dimension is to evaluate, in a generic way, any type of technique, method, or tool, which is used discover new knowledge in OSN. In the literature, there are several works reviewing the different techniques and algorithms to extract knowledge that are usually applied in OSN. Taking these works into account, we can generate an overall taxonomy of the different functions and methods that SNA tools usually provide. Then, using this taxonomy, it would be possible quantifying the **degree of value** that each tool provides, according to the functionalities that it covers within the taxonomy generated. analyzing the existing research works focused on carrying out a detailed review of the applications and methods to extract knowledge from OSN, the work presented in the 90s by Wasserman and Faust [51] is one of the most relevant publications on the area over the years. The authors show in this review the following three aspects for extracting knowledge in SNA field:

- The application of *Balance theory* is one of the main concepts to emerge from the early days of SNA [27]. It is focused on the cognition or awareness of sociometric relations, usually positive and negative affect relations such as friendship, liking, or disliking, from the perspective of an individual. The idea of balance arose in a study introduced by Fritz Heider [391] about the individual's cognition or perception of social situations. Heider was focused on a single individual and was concerned about how this individual's attitudes or opinions coincided with the attitudes or opinions of other 'entities' or people. The entities could be not only people, but also objects or statements for which one might have opinions. This author considered ties, which were signed, among a pair or a triple of entities.
- The use of *Graph theory* [53] in OSN allow us to identify the most influential, prestigious, or central actors within an OSN. These type of relevant actors are usually located in strategic locations within the network, and there are several centrality measures [29, 392] in graph theory that measure of how the position of an actor is within the overall structure of the OSN, such as the degree, betweenness, closeness, and eigenvector centrality, amongst many others (see Sections 2, 3.1.1). In addition, SNA are naturally transitive, which means that friends of a given actor can also be friends. This property of transitivity is quantified in graph theory by the clustering coefficient defined in Watts and Strogatz work [35].
- The identification of *Cohesive Subgroups or Communities* [36, 169, 88] within an OSN is one of the major concerns of SNA. Cohesive subgroups are subsets of actors among whom there are relatively strong, direct, intense, frequent, or positive ties. The concept of social group as used by social and behavioral scientists is quite general, and there are many specific properties of an OSN that are related to the cohesiveness of subgroups, so there are many possible OSN subgroup definitions (see Sections 2, 3.1.2).

Based on these main aspects, different methods and approaches have been designed and developed to discover knowledge, or extract patterns from OSN using data mining techniques. analyzing different publications reviewing the applications and methods to extract knowledge from OSN [393, 394, 395, 396, 397], in general terms, it could be

considered that the taxonomy of the main functionalities for discovering knowledge which can be embedded in SNA tools are the following:

1. *Qualitative and quantitative/statistical analysis ($F_{Value(1,i)}$)*:
   (a) *Computation of measures based on the topology* ($F_{Value(1,1)}$) of the network that provides a local and global description of it. These type of measurements are extracted from the graph theory, being some of the most relevant the density, distance, centrality or transitivity, amongst others. A huge number of works [29, 35, 392], used previous measures to discover the most influential, prestigious, or central nodes (or actors) within an OSN. The value of this measure, $F_{Value(1,1)}$, will be calculated according to the following scale: +1/5 until 1 for each measure employed (diameter, mean degree/distribution, cluster coefficient, connected components, transitivity, triangle count, etc...).
   (b) *Link analysis* ($F_{Value(1,2)}$) algorithms arise with the aim of finding the most valuable, authoritative or influential node (e.g., a webpage in the Web), being the HITS [398] and the Google Pagerank [399] the most popular ones. This measure will be calculated according to the following scale: +1/4 until 1 for each algorithm provided.
2. *Pattern Mining* methods ($F_{Value(2,i)}$):
   (a) *Community detection* ($F_{Value(2,1)}$) algorithms (static and dynamic) [36, 88, 114] try to find groups of nodes (users) where the set of edges is dense within the group and sparse outside it. One of the main difficulties in this topic is how a community is defined, since it can depend specifically on the domain where it is applied and what the network represents, as well as the type of links that are considered. Naturally, community detection algorithms are based on concepts from graph theory and clustering. Indeed, this problem is very similar to the problem of graph clustering. Recently, the analysis to the dynamics and evolution of OSNs has grown hugely, therefore many of the classic community search algorithms have been extended to study also the behavior of the communities over time. The scale to compute this measure is the following: +1/3 if allows one kind of community detection (overlapping or non-overlapping); +1/3 if allows to do overlapping and non-overlapping community detection; +1/3 if allows to do community detection in temporal networks.
   (b) *Opinion Mining* ($F_{Value(2,2)}$) techniques [67] are focused on the the detection of user opinions, and also feelings or reactions of people about certain beliefs, products, decisions or events. OSN are online sites where people can express their ideas and opinions, exchange knowledge and beliefs or criticize products. Millions of new posts giving opinion on products and services are generated every day in OSN. All this information dumped on OSN mostly in text format is very valuable to discover new knowledge. One of the most famous methods within these techniques is the *Topic Detection* algorithms, that are based on the idea of applying data mining techniques to detect what topics are more popular over the time [400]. In addition, *Sentiment analysis* methods [401] try to identify how people feel about a specific topic. This issue can be as important as detecting the topic itself on the OSN, and it can be addressed using data mining techniques related to NLP [402]. This aspect is measured according to the following scale: +1/2 if it provides topic detection; +1/2 if it provides sentiment analysis.
   (c) *Homophily Models* ($F_{Value(2,3)}$) [33]. Opinions of influential users on OSN often are capable of affecting the decisions of other users. The concept of Homophily refers to how the behavior of users that depict the same opinion (are linked under the same social community), is subject to adjustments in awareness of the behavior of other participants in the same community. The models of this concept are based on the idea that at this stage, the decision of influential participants, who are either efficient in the field or in communications skill, attracts the follower of the minority. So, when cogent information is introduced by these influential users, their opinions are disseminated in cascade affecting the positive or negative decisions of other users. In order to calculate this measure, the following scale is used: +1 if available.
3. *Predictive analysis* ($F_{Value(3,i)}$):
   (a) *Propagation and Virality Modeling* ($F_{Value(3,1)}$) consists on the study of the spread of influence through OSN. This issue has a long history in the area of social sciences, where the first studies was emerged on medical and agricultural research areas [403, 404]. In recent years, these type of models have been applied by marketing researchers, trying to model the 'word-of-mouth' diffusion process for viral marketing applications [174, 405]. The value of this measure will be calculated according to the following scale: +1 if available.
   (b) *Link Prediction* ($F_{Value(3,2)}$). In dynamic or temporal networks, a typically problem addressed is estimating the probability of two particular nodes are to become connected in the future. This is a

classical computational problem underlying in OSN evolution over time, that was introduced by Liben-Nowell and Kleinberg [406] as the link prediction problem. It can infer new interactions among members of an OSN that are likely to occur in the near future. This measure will be computed according to the following scale: $+1$ if available.

As mentioned above, this proposed taxonomy can be used to quantify the **degree of value** ($d_{Value}(t)$) that each SNA tool (t) provides according to the functionalities that it covers as shown in equation 3. Several weights are used (α, β, and γ), to represent the importance given to each characteristic. In this work, all of the value characteristics will have the same weight (so α, β, and γ will be set up to $1/3$). This equation has been normalized in the range [0,1] taking into account the different methods, techniques, algorithms and measures that are incorporated by the particular tool ($d_{Value}(t) \in [0, 1]$).

$$d_{Value}(t) = \alpha \cdot \frac{\sum_{i=1}^{2} F_{\textit{Value}(1,i)}(t)}{2} + \beta \cdot \frac{\sum_{j=1}^{3} F_{\textit{Value}(2,j)}(t)}{3} + \gamma \cdot \frac{\sum_{k=1}^{2} F_{\textit{Value}(3,i)}(t)}{2} \tag{3}$$

6.2 Scalability (D_2)

Currently, the exponential growth of data has created serious problems for traditional data analysis algorithms and techniques (such as data mining, statistics, machine learning, and so on) to processing the data available in electronic sources. A new generation of algorithms and frameworks is currently being developed in order to manage big data challenges, and they require high scalability in both memory consumption and computational time [17]. Therefore, this dimension will be used to define, and quantify, the scalability capacity of a tool or technique (e.g., algorithm) used in an OSN. From this perspective, the amount of information handled will be the key feature considered (mainly using the quantity of nodes and edges that can be processed by the SNA method or tool). A highly scalable software would work correctly on a small dataset as well as working well on a very large dataset (say millions, or billions of nodes and edges). In general terms, scalability refers to those techniques that ensure that some quality of service is maintained as the size of the data set to be managed grows, or the complexity of the addressed problem increases.

Big Data systems like MapReduce [407], Hadoop [408] or Spark [409] have been developed as a response for these scalability problems. However, the specific application areas of SNA are usually modeled as graph-theoretical problems, and unfortunately, the direct application of graph algorithms in these big data environments is often not an efficient solution (most of classical graph-based problems are NP-hard). Another approach to tackle this problem is the graph-parallel systems for specialized graph processing problems. These systems perform better than the general tools for Big Data, but their main disadvantage remains that they can only be used for graph specific problems [410, 411]. An efficient way to address this problem is to combine the advantages of both approaches: the graph-parallel approach, and the general big data processing tools. For example, GraphX is an Apache Spark's built-in library for graph analytic and graph-parallel computation [412, 413], which provides an excellent, and highly scalable, solution for graph-based problems.

In general terms, scalability is a desirable aspect of a network, system, application, or process. This concept can be defined as the capability of a system to handle an increasing number of elements or objects, to process increasing volumes of work adequately, and to be easily enlarged or extended [414]. This means that the application or system should have the ability to continue functioning correctly when the problem it is changed in size or volume, taking full advantage of it resources in terms of performance. The scalability of a system usually depends on the types of data structures and algorithms used to implement it, or if it has different components or modules (and how them have been designed), or on the communication mechanisms used by its components. For example, the data structures of a system affect not only the amount of space required to perform a particular function, but also the time. Through this last observation two of the main aspects of scalability can be appreciated: *space* and *time*. In Bondi [414] work, a more detailed analysis of scalability aspects is presented, where the author considers four main types of scalability on a system:

1. *Load* scalability: the ability to function with agility (without undue delay, without unproductive consumption of resources, and making good use of the available resources).
2. *Space* scalability: the memory requirements do not grow to intolerable levels, as the number of items that the system supports increases.
3. *Space-Time* scalability: the ability to continue operating with agility as the number of objects or data to be processed increases by orders of magnitude.
4. *Structural* scalability: a system is structurally scalable if its implementation does not impede the growth of the number of objects or items it is capable of handling.

Load scalability may be improved by exploiting parallelism, but the other three characteristics mentioned of scalability are inherent to the architectural design and implementation of the system (such as the length or the choice of data structures), and in many cases are difficult or even impossible to change. In addition, when a taxonomy of characteristics is defined, it is natural to study whether there are dependencies between them. In this particular taxonomy, for example, systems with poor space scalability or space-time scalability, might have poor load scalability, due to the attendant memory management overhead, or search costs. On the other hand, systems with good space-time scalability because their data structures are well engineered, might have poor load scalability due to poor decisions about scheduling, or parallelism, which have nothing to do with memory management.

Another taxonomy of characteristics related to the scalability was proposed by Hesham and Mostafa [415], in this particular case, it was defined for those architectures that allow parallel processing. This work presents multiple dimensions to measure the scalability, such as:

1. *Administrative* scalability: The ability for an increasing number of organizations or users to access a system.
2. *Functional* scalability: The ability to enhance the system by adding new functionality without disrupting existing functions.
3. *Geographic* scalability: The ability to maintain effectiveness during expansion from a local area to a larger region.
4. *Load* scalability: The ability to expand and contract to accommodate heavier or lighter loads.
5. *Generation* scalability: The ability of a system to scale by adopting new generations of components.
6. *Heterogeneous* scalability: The ability to adopt components from different vendors (or others environments or systems).

As can be seen, there are several works that try to describe which characteristics should be taken into account when analyzing if a system is scalable. However, any of these mentioned works provide a model to quantify the degree of scalability of a system. The Universal Scalability Law (USL) presented by Gunther [416], provided a quantification model for scalability of the systems or applications. This model is defined in terms of three main parameters α (contention), β (coherency), and γ (concurrency), that can be identified respectively with the three Cs [417]:

- *Concurrency* (γ): the maximum throughput (the measure of a number of requests processed over a unit time by the application) attainable at a given level of load.
- *Contention* (α) queuing time for shared resources.
- *Coherency* (β) delay time for data to become consistent (or coherent), by virtue of point-to-point exchange of data between resources that are distributed.

Defining the system throughput $X(N)$ at a given load, N, the USL can be expressed as the equation:

$$X(N) = \frac{\gamma \cdot N}{1 + \alpha \cdot (N-1) + \beta \cdot N \cdot (N-1)} \tag{4}$$

The independent variable N represents the number of users or data load that is incremented on a fixed hardware configuration. When the scaling is linear-rising (the case for ideal parallelism), the $\alpha = \beta = 0$. In other words, the overall throughput $X(N)$ increases in simple proportion to N.

Following the same approach of using throughput to measure scalability, in the work presented by Jogalekar and Woodside [418], the scalability is measure based on the 'power' metric of Giessler et al. [419] as follows:

$$Power = \frac{\gamma}{T} \tag{5}$$

where γ is throughput and T is mean delay time.

As mentioned at the beginning of the section, the technologies developed for big data environments have arose to address the problem of scalability due to the exponential data growth that has occurred in recent years. Nevertheless, how to measure scalability in Big data environments is a question that is still being addressed. In Sanchez et al. [420] work, the isoefficiency model is introduced as a standard measure of scalability. The isoefficiency function determines the ease with which a parallel system can maintain a constant efficiency and hence achieve speedups increasing in proportion to the number of processing elements [421]. A small isoefficiency function means that small increments in the problem size are sufficient for the efficient utilization of an increasing number of processing elements, indicating

that the parallel system is highly scalable. However, a large isoefficiency function indicates a poorly scalable parallel system.

Taking into account the different taxonomies of scalability characteristics shown by the different studies mentioned above, and adapting them specifically for SNA methods and techniques, the following sets of measures are proposed to quantify the **degree of volume**, or scalability, for SNA:

1. *Space-Time* ($F_{Volume1}$): maximum number of elements (nodes and/or edges) it is able to process without degrading its performance. The value of this measure will be calculated according to the following scale: *Low* (1/3) if process networks with less than 10.000 of nodes and/or edges; *Medium* (2/3) if process networks with nodes/edges between 10.000 to 100.000; and *Large* (1) if process networks with more than 100.000 nodes/edges. These values have been set to indicate the maximum size of the elements that can be processed by the tool or algorithm (low, medium, high). Of course, these values would be strongly modified in the coming years as processing and storage capacities increase.
2. *Parallelism* ($F_{Volume2}$): capacity of parallel computing. The value of this measure will be calculated according to the following scale: *Low* (1/3) for single or centralized processing; *Medium* (2/3) for distributed processing; and *Large* (1) for parallel processing using Big Data technologies.
3. *Functional* ($F_{Volume3}$): ability to enlarge the system or application by adding new features or extending existing ones. The following scale allow measuring this characteristic: *Low* (1/3) if none of the existing functionalities can be modified or new ones added; *Medium* (2/3) if some existing functionalities can be added or modified; and *Large* (1) if any of the existing functionalities can be added or modified.
4. *Heterogeneous-Integration* ($F_{Volume4}$): ability to integrate or communicate with components or modules from different environments or systems. The value of this measure will be calculated as follows:
   - *Low* (1/3) if cannot be integrated with components or modules from other environments or systems;
   - *Medium* (2/3) if can communicate with some components, but not all, or modules from other environments or systems;
   - *Large* (1) if can be fully integrated with components or modules from other environments or systems.

Previous metrics have been combined to generate a scalability degree (d_{Volume}) for any tool (t) as it is shown in equation 6, where scalability ($d_{Volumne}(t)$) is rated from 0 to 1 depending on its capacity to scale to large data sets ($d_{Volume}(t) \in [0, 1]$).

$$d_{Volume}(t) = \frac{\sum_{i=1}^{4} F_{Volume_i}(t)}{4} \tag{6}$$

6.3 Information Fusion & Integration (D_3)

OSN popularity is increasing everyday thanks, in part, to the diverse services provided and the target groups that can use them. In this way, it is possible to find OSN for any purpose ranging from personal (such as Facebook, or Twitter), to the professional ones like ResearchGate, or LinkedIn among others. Also these OSN offer a wide variety of services, and it is possible to share photos (Flicker, Instagram, ...), videos and music (Facebook, Youtube, ...), or micro-blogging (Facebook, Twitter, ...), etc. In addition to the service provided by the OSN, all of them allow different kind of interactions between the users (for example in Twitter users can '*follow*' other users, whereas in Facebook they are '*friends*'), and different actions over the content published by the user, i.e., in any OSN anyone can '*like*', '*comment*', and '*share*' the contents published by other users.

With all of this, it is easy to realize the amount, and diversity, of data available in the different OSN that can be used to perform SNA tasks. In order to measure this diversity, we define this dimension called *Information Fusion*, or *Integration*, that tries to answer the question: "*What kind of data can I integrate?*. This measure would be equivalent to the concept of "*Variety*" from the Big Data paradigm. In the case of OSN, this dimension will measure different aspects regarding the data used to perform the SNA tasks. In this case, we have defined three different measurements that will take into account: (1) the number of different type of data (*multichannel*), (2) the number of different OSNs used to extract the data (*multimodality*), (3) and the representation of this data into the model (*multi-representation*).

Following, we provide a detailed description of these three measures and also, a formal definition to measure them:

1. *Multichannel* (F_{Var1}): this indicator measures the diversity of the data taking into account the format of the data. In this regard, SNA algorithms are able to extract knowledge by using data from two different sources of

information: the graph resulting from the modeling process, and the content published in the OSN [422]. In this sense, the different types of data can be used to perform SNA tasks, are the following:

(a) Edges of the graph: this data source is related to the interactions of the different users in the SN.
(b) Text: in this case, the data used to perform SNA tasks is the content published by the users in text format, for example *tweets*, *comments*, etc.
(c) Images: this information is extracted directly from the images, as photos or memes, that users post in any OSN. This data could be the direct analysis of the photo, or the tags that describe the content of the photo, etc [343].
(d) Video: in any OSN is possible to publish videos, and the different SNA algorithms can take advantage of the video content to extract some valuable information.

To sum up, F_{Var1} can be defined as the number of different data formats handled by the algorithms or tools. As it is quite difficult to fix the number of data formats due to the evolution of OSN, in this work we are going to limit the value of this characteristic depending on the different number of data types used. In this sense, we are going to consider three different ranges of values for F_{Var1} which are:

- *Low*: this value reflect the case where the algorithm uses one type of data, i.e: only the edges of the graph, or text, or images, etc. When the algorithm, or system, uses only one type of data, it will have low M_c and its value is $1/3$.
- *Medium*: this value considers the cases where the algorithms, or systems, integrates 2 or 3 different types of data. The value of M_c in this case is $2/3$.
- *High*: when the systems, or the algorithms, are able to integrate 4 or more different types of data, the value for its M_c will be 1.

For this reason, $M_c \in [1/3, 1]$. It is important to note that this indicator is independent of how data source is modeled, or what kind of information is extracted. Imagine a platform that uses the tweets in text format, the M_c value for this format will be $1/3$ independently of the analysis performed with this data.

2. *Multimodality* (F_{Var2}): refers to the different number of data sources (independently of the data format) that can be handled by the algorithm. This indicator takes into account the different number of different OSN data formats, which are integrated by the algorithm. In this sense, it is quite difficult to define the limits of this indicator due to the fast evolution of OSN (theoretically this value would be a positive natural value, $\mathbb{N}^+$). Some examples of OSN are: Facebook, Twitter, Instagram [343], Youtube, or LinkedIn, to mention just a few of them.

   For this reason, Multimodality is defined in the same way as *Multichannel*, i.e., using three different values that evaluate the different number of OSN taken into account:

   - *Low* ($F_{Var2} = 1/3$): this value represents the systems, or algorithms, that gather data from one OSN.
   - *Medium* ($F_{Var2} = 2/3$: it is used when the data used to perform SNA tasks are extracted from 2 or 3 different OSN.
   - *High* ($F_{Var2} = 1$): this value is used when the system, or algorithm, is able to integrate data from 4 or more OSN.

   With all of that, $M_m \in [1/3, 1]$, where $1/3$ means that only one OSN has been used to extract the data, whereas 1 means that the 4 or more OSN have been taken into account. Note that the value of this indicator is not ranking the different OSN but only providing a metric about the number of different OSN used by the platform/algorithm. For example, the M_m value for a platform that is able only to extract data from Youtube will be $1/3$; and the value for other platform that integrate data from Twitter and Instagram will be $2/3$.

3. *Multi-representation* (F_{Var3}) provides a quantitative measure of the data model representation used by the algorithm, and its value will depend on the complexity of the representation model. It is defined as $M_r \in [1/3, 1]$, where a value of $1/3$ means the *basic* representation model, whereas a value of 1, the most *advanced* one. Following, we explain the different representation levels with their corresponding values:

   (a) *Basic model*: this kind of representation correspond to the case when the OSN is modeled into a simple unweighted graph $G = (V, E)$. In this graph, V is the set of nodes of the graph and represents the set of users of the SN; whereas E corresponds to the set of edges of the graph and represents some sort of connection between the users. Note that depending on the OSN taking into account this graph could be directed or undirected. The value of F_{Var3} in this case is the lower one, i.e., $1/3$.
   (b) *Intermediate model* corresponds to the value of $2/3$. In this case, the problem taken into account is modeled into a simple weighted graph. The resulting graph is quite similar to the previous representation but the edges contain a value providing some kind of information, for example frequency of a specific action, or a meaning of the relation.

(c) *Advanced model*: this level represents the most advanced model used to work with the data extracted from the OSN. In this case, we consider a multi-layer SNA representation as the most advance model. *Multi-layer networks* considers multiple channels of connectivity, it is a representation used to describe systems where the different actors are interconnected by different categories of connections. In this kind of networks, each channel (i.e., each type of relation) is represented by a layer, and the same node may have different set of neighbors in different layers. The value of F_{Var3} when the data is modeled into a Multi-layer network is 1.

Imagine a system that is able to analyze three different types of interactions (F_{Var_i}) between users in an OSN. Fig. 6 shows the different representations taken into account by the multi-representation measure to model this situation. The first one (Fig. 6.a) represents all the information into a unweighted graph, i.e., the *Basic model*. Also, it is possible to use the *Intermediate model* with a weighted graph (Fig. 6.b). In this case, the weighted can be a number to represent, for example, the frequency of an interaction or a label to differentiate the different interactions. Finally, the *Advanced model* is shown in Fig. 6.c, where the data is modeled into a multi-layer graph and each graph contains the information regarding each kind of interaction.

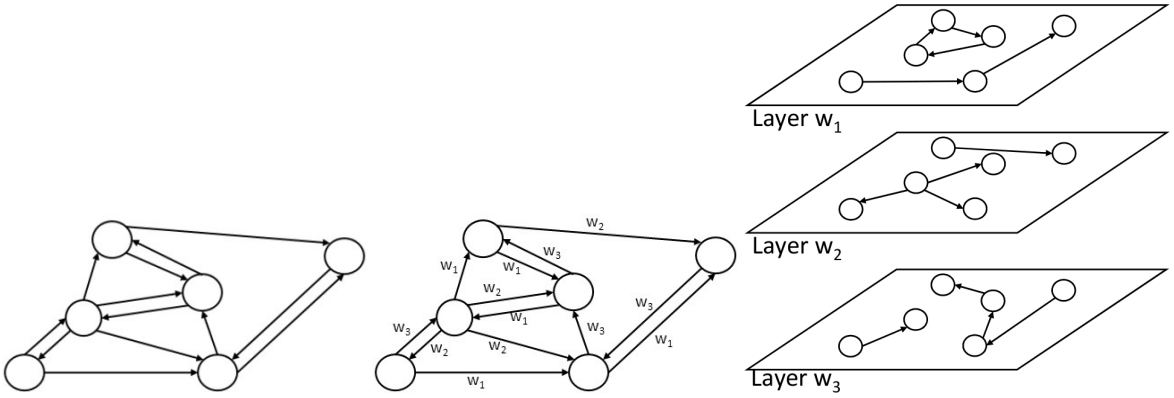


Figure 6: This figure shows the different representations taken into account into the F_{Var_3} dimension. The different representations are: a) Basic model, b) Intermediate model, and c) Advanced model

The next step is to define the value of the Information Fusion/Integration dimension, i.e., the **degree of variety**, in terms of the three measures just explained. From these three indicators, we consider that *Multimodality* (F_{Var_2}) and *Multichannel* (F_{Var_1}) play an important role in terms of Information Fusion/Integration because both measures describe, respectively, the number of different OSN analyzed and the number of different data format taken into account. For this reason, this dimension can be defined as:

$$d_{Variety}(t) = \frac{\sum_{i=1}^{3} F_{Var_i}(t)}{3} \tag{7}$$

Following Eq. 7, those algorithms, and tools, using different types of data extracted from different OSN will perform higher than others using different types of data extracted from the same OSN. Finally, the fraction is used to normalize the value of $d_{Variety}$, thus this dimension is assessed using this degree in a range of [0,1] ($d_{Variety}(t) \in [0, 1]$).

6.4 Visualization (D_4)

What can the system show from an OSN is one of the key issues for any researcher, analyst or end-user that works in the domain of SNA. The concept of Visualization is used as a dimension to measure the capacity of the tools, frameworks, and methods to visually represent the information stored in the network [20]. Due to the human brain, it is easier for everybody to visualize large amount of data instead of reading tables or reports. For this reason, data visualization is a quick and easy way to convey concepts in a universal manner. In order to create good visualizations, one must first decide which questions want to be answered and select compelling visual encodings that depict the data values as graphical features such as position, size or orientation. Although the amount of possible visualization designs is extremely large, statisticians, psychologists, and computer scientists have studied the most suitable representations for different types of data, easing the difficulty of choosing a proper visual encoding. Regarding visualizing OSNs, the most common diagrams found in the literature are: *node-link* and *matrix* diagrams [423, 424]. Although, lately, new

alternative diagrams have been proposed that fall outside any of these two categories [425, 426, 427]. An example of these diagrams can be found at Fig. 7, where a node-link diagram (Fig. 7.a), a matrix diagram (Fig. 7.b) and the novel HivePlot diagram (Fig. 7.c) are shown.

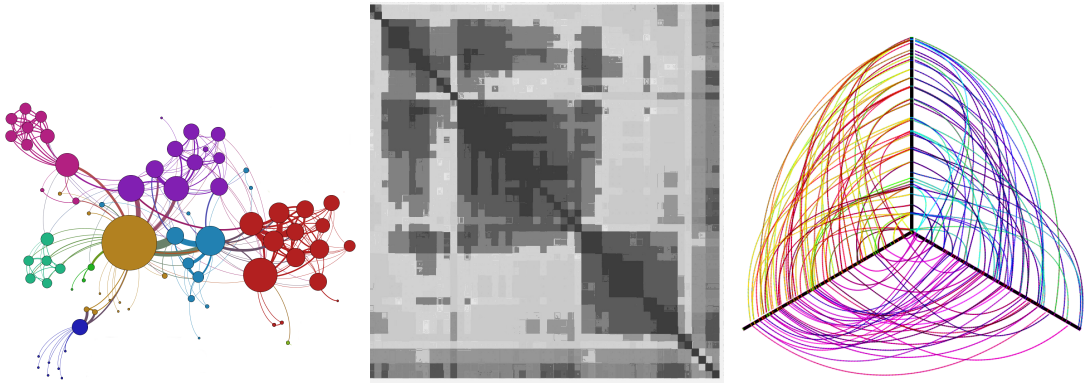


Figure 7: Subfigure a) shows an example of a node-link diagram; Subfigure b) shows an example of a matrix diagram; and Subfigure c) shows an example of an alternative diagram named HivePlot [425].

The most common visual representation of an OSN is the node-link diagram (see Fig. 7.a), where individuals of the network are represented as dots, and a relation between two individuals is represented by an arrow connecting them. Depending on the relationship, the edge connecting two nodes can be directed or not. Placing the individuals in such a way that the resulting diagram is legible is not a trivial task and many efforts have been made to ensure this. Since the 80's many publications have addressed this task and a plethora of methods have been proposed [428, 429, 430, 431, 432, 433]. The main approach followed so far considers the network as a physical system where forces are applied to the individuals moving them around a 2D or 3D plane [434], some examples are the Spring-Electrical or the Stress and Strain models. In the former (Spring-Electrical models), individuals are simulated as positive charges and connections between them as springs. First, all users are allocated in a random position. Then, the physical simulation takes place, individuals repels other individuals around them and the springs avoid separating connected individuals too far apart. The simulation continue for a predefined number of steps or until the system stabilizes. The most famous algorithm is this category is the Fruchterman-Reingold [429]. In the latter, the Stress and Strain models, the positive charges are discarded and only the springs are used. These models, first define the desired spring length. Then, the individuals are allocated at random positions. Contrary to the spring-electrical model, there are no simulation and an "imbalance degree" is defined as the difference between each spring length and the desired spring length. Finally, the "imbalanced degree" is minimized changing the individual's positions. The most famous algorithm in this category is the Kamada and Kawai [430]. Node-link network visualization is an active research field and a wide variety of methods have been proposed to minimize their computational requirements, or to improve their output on bigger networks [431, 432, 433].

Although the node-link diagram with the physical systems simulations works very well for a wide number of different networks, they usually have trouble visualizing small-world networks [35]. In a small-world networks, any two nodes are not very far from each other (i.e., there are short path lengths). For that reason, the use of node-link diagrams with the physical system analogy only produces a mess similar to a hairball where it is impossible to see anything. This is a big problem in the area of SNA as, usually, OSN are also small-world networks. Thus, to solve this shortcoming, some alternative representations have been proposed, like the matrix diagrams (see Fig. 7.b). These diagrams represent the networks by their adjacency matrix. The adjacency matrix of a network has all its users as rows and columns, and the value of each cell represents the strength of the connection between the two corresponding individuals. In a matrix diagram, the adjacency matrix of a network is drawn as a picture, where each cell of it is represented as a colored square depending on its strength. This representation has its advantages and disadvantages. On the one hand, all the connections are always visible, regardless of the network size, in the node-link diagram connections cross one another and, sometimes, it can be impossible to distinguish them. On the other hand, identifying paths in the network is harder than ordering the rows and columns of the adjacency matrix, this process is called *Seriation*, which is essential to unveil the inner structure of a network. *Seriation* is not a trivial task, in fact, it is an active research area where a large number publications are coming out to tackle this problem [435].

There is not a consensus about what is the best strategy for visualizing networks as both of the aforementioned approaches have their advantages and disadvantages. In addition, some hybrid approaches have also been proposed like, MatrixExplorer [436] or Matlink [437]. Finally, within this debate, there are some scientific groups that are proposing alternative diagrams, tailored for specific areas, that deviates from the mainstream. For example, in the Biology area,

we can find: HivePlots [425], where nodes are mapped to and positioned on radially distributed linear axes and edges are drawn as curved links; or BioFabric [426] where nodes are depicted as horizontal lines, not as points. Another example is PivotGraph [427], that is specialized in multivariate data.

How the network is represented is not the only characteristic used to compare SNA visualization tools, but there are other characteristics that need to be taken into account [434, 438, 439]. These characteristics are: "volume", "summary" and "interaction". The first one, volume, asses the raw processing capacity of a tool, how many users/nodes and connections/edges can be handled by the tool. Although the capacity of visualizing millions of nodes does not imply the generation of outputs in such a way any human can use. Most of the times, visualizing a medium-size network, without any extra help, provides as a result an unreadable hairball. In this situation the second characteristic, name "summary" becomes useful. "Summary" evaluates the capacity of a tool to generate simplifications that allows to reduce the complexity of the network while maintaining as much information as possible. This balance, between the summary and the raw data, must not be static and should adapt to the user's requirements. The third and last characteristic, "interaction", measures the capacity of a tool to adapt its graphical output to the user's needs. These three characteristics together allow us to fulfill the Shneiderman's visualization mantra: "Overview first, then zoom and filter details on-demand" [440]. Therefore, these three characteristics have been considered as the essential ones that any SNA visualization tool should have.

Although these three characteristics are great for describing and comparing SNA tools, two of them, "volume" and "summary", are highly correlated with the other three dimensions presented in this work (Scalability, Pattern & Knowledge discovery, and Information Fusion & Integration). Therefore, we have decided to move away from the approach used in the literature and remove any aspect not related to graphics from the visualization dimension. Hence, we will describe the visualization dimension in a twofold way. On the one hand, we will use the aforementioned "interaction" characteristic. On the other hand, the lack of consensus over what is the most suitable diagram to visualize an OSN makes impossible to use it as a comparison method. Thus, we have considered that all the aforementioned diagrams are equally valuable when visualizing an OSN. Therefore, we will introduce an extra characteristic, named "*Visual Variables*", that will help us to asses the information representation capacity of a tool. *Visual Variables* were proposed in [441] by Jacques Bertin based on his experience as a cartographer and geographer. In his work he described the *visual variables* as the fundamental way in which graphic symbols can be distinguished. The author identified the Visual Variables listed below and, according to his work, it can be used for four different purposes: (1) Selective (easily distinguish between groups); (2) Associative (identify changes among the same group); (3) Ordered (allows to identify sequences); and, (4) Quantitative (allows to compare numerical values). The visual variables can be summarized as follows:

1. **Position** (F_{VisVar_1}): refers to the location of an object in the image, position is one of the most versatile variable as it can be used in as a Selective, Associative, Ordered or Quantitative variable. For example, if an OSN has geo-localization data, the position of an user can display his/her country.
2. **Size** (F_{VisVar_2}): refers to the size variation of an object. It can be used as: Selective, Ordered or Quantitative variable. For example, changing the size of an user node depending on his/her degree.
3. **Shape** (F_{VisVar_3}): refers to the different geometrical shapes an object can have (triangles, rectangles, circles... etc.). It can only be used as a Associative variable. For example, showing users from different OSN with a different shape.
4. **Orientation** (F_{VisVar_4}): refers to the rotation an object presents. It can be used as a Selective or Associative variable. For example, using an arrow to indicate the direction of a following/follower relation.
5. **Color** (F_{VisVar_5}): refers to the color hue of an object. It can be used as a Selective or Associative variable. For example, using the same color for all the users in the same community.
6. **Saturation** (F_{VisVar_6}): refers to color saturation, the brighter or lighter a color hue is. It can be used as a Selective, Ordered or Quantitative variable. For example, showing users with a higher centrality colored with a more saturated color.
7. **Texture** (F_{VisVar_7}): refers to the fill pattern of an object. It can be used as a Selective or Associative variable. For example, using two different patterns to identify members of different political parties.

In our opinion previous variables are suitable to act as measures of the visualization dimension. Instead of trying to evaluate the quality of a visualization method by the type of representation used, we will evaluate it by the number of visual variables used by a tool to enrich them. The more visual variables a tool is able to handle, the more extra information it can visually represented, and the higher the visualization degree will have the tool. However, as it has been stated by the literature, being able to represent a lot of information does not implies the generation of good quality visualizations. That is why the "interaction" capabilities of a tool are also being considered when calculating a

visualization degree of a tool. The most common interactions analyzed in the literature can be summarized with the next five actions:

1. **Zoom** (F_{Inter_1}): refers to the action of changing the level of details of the elements being shown but maintaining the same number of them. This does not implies that all elements need to be visible in the screen at once. For example, using the mouse wheel to zoom in/out a graph.
2. **Filter** (F_{Inter_2}): contrary to zoom, refers to changing the number of elements being displayed but maintaining the same level of detail. For example, dragging the mouse to pan over a diagram.
3. **Highlight** (F_{Inter_3}): refers to the action of emphasizing some particular elements of a set. For example, highlighting the neighbors of an user when the mouse is positioned over it.
4. **Grouping** (F_{Inter_4}): refers to the action of replacing a group of elements with a simplification that maintains all or some of the properties of the group. For example, joining all the users of the same country into a single node.
5. **Multiview** (F_{Inter_5}): refers to the action of switching between multiple representations of the same data. For example, switching between the node-link and matrix representation of an OSN.

In order to generate a single value capable of evaluating the degree of the visualization dimension, Equation 8 is proposed. Where F_{Visual} is the number of visual variables a tool can handle, and F_{Inter} is the number of interactions available on a tool, whereas α, β, γ, θ are weights that represent the importance given to each characteristic.

$$d_{Visual}(t) = \alpha \cdot \frac{\sum_{i=1}^{7} \gamma_i \cdot F_{VisVar_i}(t)}{7} + \beta \cdot \frac{\sum_{j=1}^{5} \theta_j \cdot F_{Inter_j}(t)}{5} \tag{8}$$

In this work, all of the visual characteristics (visual variables and interactions) will have the same weight (so α and β will be set up to 0.5), and all of the features for each characteristic will have the same importance (therefore γ and θ will be set up to 1.0).

6.5 Summary on Social Network Analysis Dimensions

Following the previous subsections, the four basic research questions (RQ) proposed in the introduction, and their related dimensions, have been mapped into a set of measures, or $SNA_{degrees}$, which can be used to provide a quantitative value for each dimension. These research questions, together with the proposed dimensions, and the metrics, or degrees, defined to assess them are shown in Table 1.

Table 1: Summary on dimensions and the quantitative measures (degrees) defined.

Research Quest.	**Dimension** (D_i)	**Degree** (d_v*)	**Range**
What can I learn?	Pattern & Know. discovery (D_1)	$d_{Value}(t) = 1/3 \cdot (\frac{\sum_{i=1}^{2} F_{Val_{(1,j)}}(t) + F_{Val_{(3,j)}}(t)}{2} + \frac{\sum_{i=1}^{3} F_{Val_{(2,i)}}(t)}{3})$	$[0,1]$
What is the limit?	Scalability (D_2)	$d_{Volume}(t) = \frac{\sum_{i=1}^{4} F_{Volume_i}(t)}{4}$	$[0,1]$
What kind of data can I integrate?	Information Fusion & Integration (D_3)	$d_{Variety}(t) = \frac{\sum_{i=1}^{3} F_{Var_i}(t)}{3}$	$[0,1]$
What can I show?	Visualization (D_4)	$d_{Visual}(t) = 1/2 \cdot \frac{\sum_{i=1}^{7} F_{VisVar_i}(t)}{7} + 1/2 \cdot \frac{\sum_{j=1}^{5} F_{Inter_j}(t)}{5}$	$[0,1]$

Fig. 8 shows an example of an hypothetical algorithm, framework and a tool, and how the proposed dimensions, can be used to represent the technology maturity to work with OSN data. We have decided to use a spider, or radar, diagram for representing the evaluation of the SNA technology. As can be seen in this figure, the axes will be used to represent each dimension, so the quality, strengths or weaknesses for each method, tool or framework can be easily analyzed.

Finally, and considering these $SNA_{degrees}$ it is quite straightforward to define a new **global metric**, that we have called $\mathfrak{C}_{SNA}$, to represent the "*Capability*" and power to work with OSN sources, to later use as a metric to rank the technologies analyzed. We calculate the value of $\mathfrak{C}i$, where i represents the number of dimensions to be considered, as

the area contained in the irregular polygon defined by the i dimensions used in our previous representation (see Fig. 8). This equation comes from the Shoelace formula, also known as Gauss's area formula and the surveyor's formula [442], and it is a simple formula for finding the area of a polygon given the coordinates of its vertices, see Equation 9, where A is the area of the polygon, n is the number of sides, and (x_i, y_i) are the vertices of the polygon.

$$A = \frac{1}{2}\left|\sum_{i=1}^{n-1}(x_i y_{i+1} + x_n y_1) - \sum_{i=1}^{n-1}(x_{i+1} y_i - x_1 y_n)\right| \tag{9}$$

Therefore, the $\mathfrak{C}_i$ metric is a particular case of previous Equation with $n = 4$, and it can be mapped as Equation 10 shows. Taking into account that all of the dimensions defined are represented in an axis, all of the vertices will have at least one of the coordinates (x or y) to 0, due all of the dimensions defined in this work have been normalized to 1, implies that $\mathfrak{C}_i \in [0, 2]$. Considering: $d_{Value} = (x_1, 0)$, $d_{Volume} = (0, y_2)$, $d_{Variety} = (x_3, 0)$, $d_{Visual} = (0, y_4)$, Equation 10 can be easily simplified as Equation 11 shows.

$$\mathfrak{C}_4(t) = \frac{1}{2} \cdot \left| \sum_{i=1=\textit{Value}}^{3=\textit{Variety}} \left(d_i^{x_i}(t) \cdot d_i^{y_{i+1}}(t) + d_i^{x_n}(t) \cdot d_i^{y_1}(t) \right) - \sum_{i=1=\textit{Value}}^{3=\textit{Variety}} \left(d_i^{x_{i+1}}(t) \cdot d_i^{y_i}(t) - d_i^{x_1}(t) \cdot d_i^{y x_n}(t) \right) \right| \tag{10}$$

$$\mathfrak{C}_4(t) = \frac{1}{2} \cdot \left| \left(d^{x_1}_{\textit{Value}}(t) + d^{y_2}_{\textit{Volume}}(t) \right) \cdot \left(d^{x_3}_{\textit{Variety}}(t) + d^{y_4}_{\textit{Visual}}(t) \right) \right| \tag{11}$$

Analyzing Equation 11, two different (extreme) cases could appear: (1) if two of the dimensions are equal to zero (i.e., $d_{\textit{Value}} = d_{\textit{Volume}} = 0$, or $d_{\textit{Variety}} = d_{\textit{Visual}} = 0$), C_4 will be equal to 0, in such case the $\mathfrak{C}_4$ must be calculated as Equation 12 shows (this is equivalent to make a projection of both non-zero dimensions into two different axis). (2) If three dimensions are zero, there's no possibility to calculate any area due there's only one dimension with a value, in such case the $\textit{SNA}_{\textit{degree}}$ should be used to evaluate the capability (only in that dimension) of the SNA technology. Fig. 8 shows an example of our Capability metric on three hypothetical frameworks).

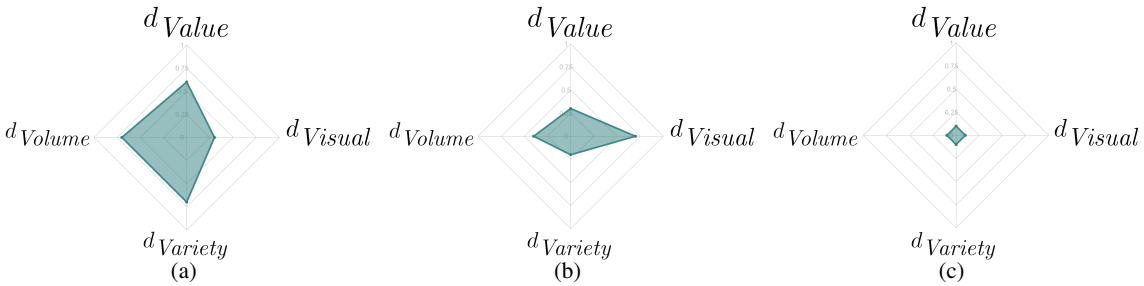


Figure 8: Spider diagrams representing the evaluation of three (hypothetical) SNA platforms by using the 4 dimensions described in this work. The exact values for each dimensions for these 3 hypothetical frameworks are shown in Table 2. The figures shown correspond to a good platform (a), an intermediate tool (b) and a poor platform (c).

$$\mathfrak{C}_4(t) = \frac{1}{2} \cdot \left| d_i^{x_i}(t) \cdot d_j^{y_j}(t) \right|, d_i, di \neq 0 \tag{12}$$

Following our previous example, the three different SNA technologies previously presented in Fig. 8 would be ranked using our metric (see Table 2). Although this comparison would not have sense (as we are comparing different kinds of technology), it shows how this general metric can be used to better understand the capability of a particular SNA technology. This new metric will be used in next sections to provide a ranking between the analyzed SNA tools and frameworks.

7 Frameworks & Tools Analysis

This section provides a review on a set of popular SNA frameworks and tools, which are extensively used by the research community and industry. Due, it would be highly difficult to analyze all of the currently available tools, we have

Table 2: Example of $\mathfrak{C}_4$ metric application over three different hypothetical SNA tools and frameworks.

Rank	$\mathfrak{C}_4(t)$	**Dimensions**				**SNA technology**
		d_{Value}	d_{Volume}	$d_{Variety}$	d_{Visual}	
1	0.315	0.6	0.7	0.7	0.3	Tool 1 (high value)
2	0.170	0.3	0.4	0.2	0.7	Framework 1 (medium value)
3	0.010	0.1	0.1	0.1	0.1	Tool 2 (low value)

selected for analyzing a subset of 20 representatives. In this set, it can be found computing libraries, web applications, distributed applications, or desktop applications among others, we will refer to all of them as *SNA-software*. Initially, a list of 70 SNA-software candidates was generated using Github[9], KDnuggets[10,11] and the Infovis-Wiki[12] websites. It is important to highlight that we have considered SNA-software with both types of licenses, open source and commercial (proprietary). From the list of 70 SNA-software candidates, a set of 20 tools was selected for analysis. In this selection, it was considered the type of software license, the quantity and *quality* of the software documentation, and its current impact between the community (taking into account the popularity of some tools in published works, websites and other technical material). Following, the list containing the 20 different SNA-software analyzed in this work is shown. For each software, we provide the name, a bibliographical reference (or a website), a brief description and also, its license type:

1. **Igraph** [443]: a collection of network analysis tools with the emphasis on efficiency, portability and ease of use. License: MIT.
2. **AllegroGraph** [444]: an ultra scalable, high-performance, and transactional Semantic Graph Database. License: Proprietary.
3. **LaNet-vi** [445]: a large networks visualization tool. It provides images of large scale networks on a two-dimensional layout. License: AFL.
4. **Stanford Network analysis Platform (SNAP)** [446]: a general purpose, high performance system for analysis and manipulation of large networks. License: BSD.
5. **ORA-LITE/PRO**furlcasos.cs.cmu.edu/index.php: a dynamic meta-network assessment and analysis tool developed by CASOS at Carnegie Mellon. It contains hundreds of OSNs, dynamic network metrics, trail metrics (path-based metrics), and procedures for grouping nodes. License: Proprietary.
6. **Network workbench** [447]: a Large-Scale Network analysis, Modeling and Visualization Toolkit for Biomedical, Social Science and Physics Research. License: open-source.
7. **NetMiner**furlnetminer.com: a premium software tool for Exploratory analysis and Visualization of Network Data. License: Proprietary.
8. **Circulo** [448]: a "Community Detection" Evaluation Framework written primarily in Python. License: Apache 2.0.
9. **Cytoscape** [449]: a software platform for computational biology and bioinformatics, useful for integrating data, and for visualizing and performing calculations on molecular interaction networks. License: LGPL.
10. **JUNG** [450]: a software library that provides a common and extendable language for the modeling, analysis, and visualization of data that can be represented as a graph or network. License BSD.
11. **SparklingGraph** [451]: a Cross-platform tool to perform large-scale, distributed network computations with Apache Spark's GraphX module. License: BSD 2.
12. **NetworkX** [452]: a Python language software package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks. License: BSD.
13. **Pajek** [453]: is a Windows program for analysis and visualization of large networks. License: Unknown.
14. **GraphX Apache Spark** [413]: a module to perform graph-related parallel computation. License: Apache 2.0.
15. **Gephi** [454]: an open-source network analysis and visualization software package written in Java on the NetBeans platform. License GPL3.
16. **UCINET** [455]: a software package for the analysis of OSN data. It comes with the NetDraw network visualization tool. License: Proprietary.
17. **Prefuse** [456]: a Java-based toolkit for the interactive creation of information visualization applications (not only for graphs, but also tables and trees). License: Unknown.

[9] `http://github.com/briatte/awesome-network-analysis`

[10] `http://kdnuggets.com/2015/06/top-30-social-network-analysis-visualization-tools.html/2`

[11] `http://kdnuggets.com/software/social-network-analysis.html`

[12] `http://infovis-wiki.net/wiki/Main_Page`

18. **Graphistry**[13]: a cloud service that automatically transforms your data into interactive, visual investigation maps built for the needs of analysts. License: Proprietary.
19. **GraphViz** [457]: a open source graph visualization software. License: CPL.
20. **Neo4j** [458]: an Open source, scalable graph database. License GPL3.

Once the list of SNA-software has been briefly described, each software has been evaluated using the different metrics proposed in Section 6. The evaluation process carried out in this work follows a "top-down" approach. First, the global capability metric ($\mathfrak{C}_4$) is analyzed for each framework and tool. In a second step, we have analyzed in detail the different dimensions (i.e., the $SNA_{degrees}$) that compose the global metric. The evaluation process for each software was carried out as follows: once the initial set of tools was selected, the authors agreed an evaluation rubric (that is publicly available, see Appendix) to assess the tool. This rubric is based on the analysis of the software documentation, their official websites (or any related site that could store relevant information), and other published works that provide technical details about these tools. From this technical documentation, we assess each of the characteristics that form the different $SNA_{degrees}$, to finally obtain a quantitative value for each of the proposed dimensions. The features used in this rubric (strictly) follows the characteristics proposed to measure the four dimensions (see Sections from 6.1 to 6.4), so from these features we can obtain a quantitative value for each degree. Although the authors have previous experience in several of the analyzed tools (such as Igraph, Circulo, JUNG, or Gephi), it is not possible to download, install, and generate experimental datasets and evaluations for each SNA-software. For this reason, we decided to carry out the evaluation of the software following the previous process.

The goal of this analysis is twofold. On the one hand, it allows us to understand the strengths and weaknesses of the different SNA-software. This analysis will help any researcher who is looking for a SNA tool to select the one that best fit to his/her requirements. The type of license used by the SNA-software (public, open-source, BSD, MIT, proprietary ,etc.) can be a determining factor for future research, or the development of new products. Therefore, our analysis will take into account this feature to differentiate between those types of software. On the other hand, the analysis of the disaggregated values allows us to understand the opportunities, and weaknesses, of the tools from the SNA point of view. In this sense, this disaggregated analysis will allow us to understand if the requirements for a specific dimension (e.g., visualization) are currently fulfilled by the SNA-software available, or if there is any specific dimension that requires from some special reinforcement. Therefore, this second analysis will help the reader to detect spaces for improvement in some particular research areas related to SNA.

Table 3: Top-5 best SNA tools under Proprietary or Open-Source, and Only Open-source, licenses.

Proprietary or Open-Source					
Tool	$\mathfrak{C}_4(t)$	d_{Val}	d_{Var}	d_{Vol}	d_{Vis}
Graphistry	0.67	0.33	1.0	1.0	1.0
Neo4j	0.57	0.48	0.56	1.0	1.0
ORA-LITE/PRO	0.56	0.84	0.67	0.5	1.0
NetMiner	0.52	0.65	0.89	0.67	0.69
Cytoscape	0.39	0.67	0.33	0.58	0.93

Only Open-Source					
Tool	$\mathfrak{C}_4(t)$	d_{Val}	d_{Var}	d_{Vol}	d_{Vis}
Neo4j	0.57	0.48	0.56	1.0	1.0
Cytoscape	0.39	0.67	0.33	0.58	0.93
Gephi	0.35	0.35	0.44	0.66	0.93
Pajek	0.31	0.48	0.56	0.50	0.73
JUNG	0.28	0.41	0.33	0.75	0.64

Regarding the *global metric* score ($\mathfrak{C}_4(t)$), Table 3 shows the top-5 "Proprietary or Open-source", and "Only Open-Source" SNA-software. In addition to the aforementioned table, Fig. 9 is presented. This figure contains the distribution of the dimension scores in order to allow the reader to contextualize the values presented. The tool that provides the best global score is *Graphistry*. This tool has the highest possible scores in *Information Fusion*, *Scalability*, and *Visualization*. However, the score on *Knowledge Discovery* is below the average, as a matter of fact, *Graphistry* is the 13^{th} tool in that category. Regarding open-source tools, the best one is *Neo4j*. This tool has an above-average *Knowledge Discovery* score, an average *Information Fusion* score, and the maximum possible scores for *Scalability* and *Visualization*. Note that the scores obtained by the tools in the top 5 differ between dimensions. This means that there is

[13] `http://graphistry.com`

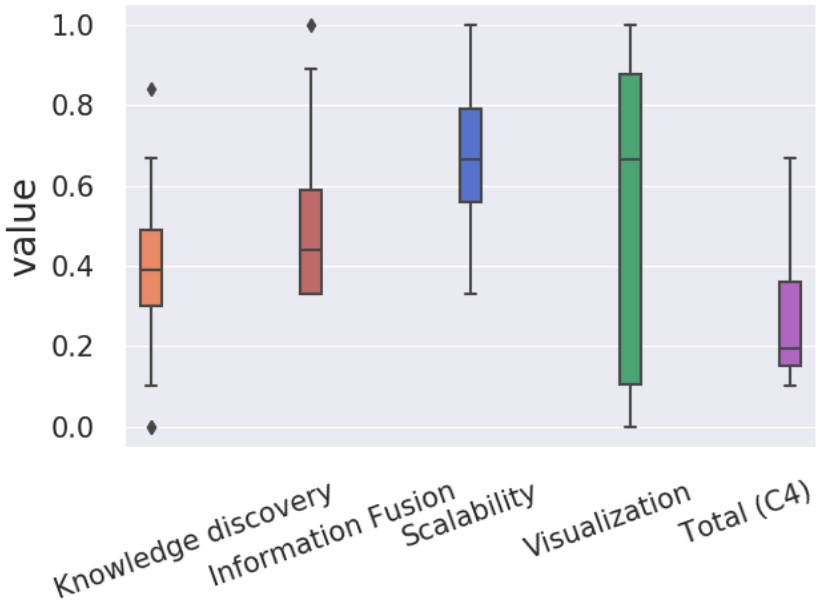
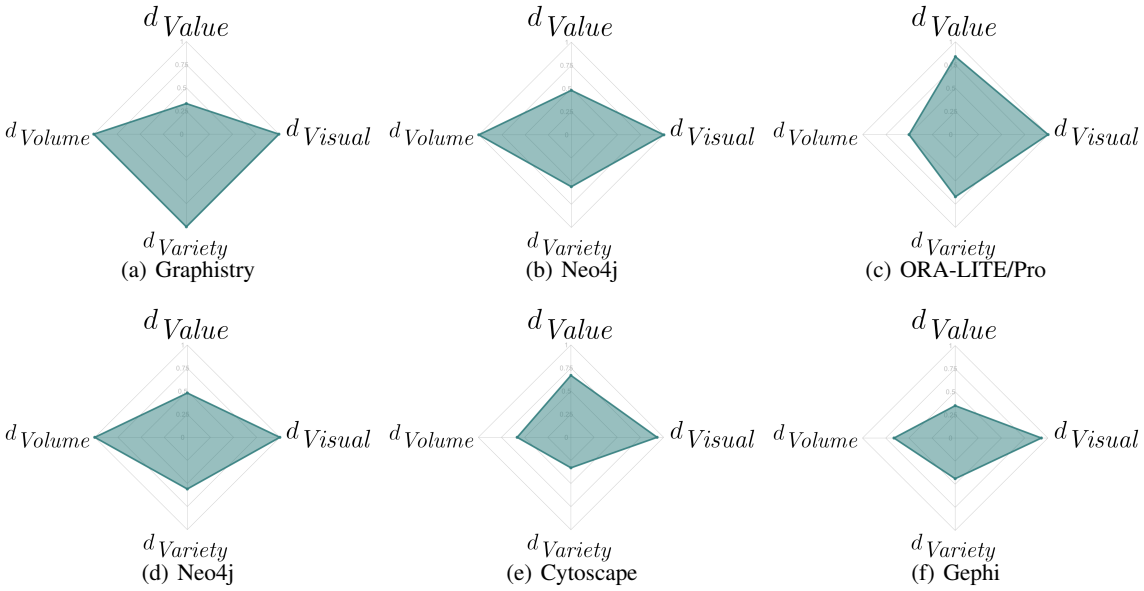


Figure 9: Distribution of the different dimensions values achieved by the analyzed tools. The X-axis represents the proposed dimensions, whereas the Y-axis shows the values obtained once the quantitative metrics (SNA_{degree}) are calculated.



(a) Graphistry (b) Neo4j (c) ORA-LITE/Pro

(d) Neo4j (e) Cytoscape (f) Gephi

Figure 10: This figure shows the top 3 SNA software with “Proprietary or Open Source" licenses (first row: Graphistry, Neo4j and ORA-LITE/Pro), and “Only Open Source" licenses (second row: Neo4j, Cytoscape and Gephi).

not a single tool that dominates all the others for all the dimensions. Furthermore, nearly all the tools analyzed, 15 out of 20, have achieved a position in one of the top 5 proposed.

Fig. 10 shows the spider, or radar, diagrams for the top-3 best SNA-software analyzed. The upper row, composed of sub-figures a, b and c, corresponds to those tools under “Proprietary or Open-Source" licenses, whereas the second row (sub-figures d, e and f) corresponds to those frameworks or tools under the “Only Open-source" license. The corresponding values for each dimension are shown in Table 3.

Table 4: Top 5 SNA-software by dimension.

Dimension	SNA-software	Score	Dimension	SNA-software	Score
Knowledge Discovery	**ORA-LITE/PRO**	**0.84**	**Scalability**	**Grasphistry**	**1.00**
	SNAP	0.67		**AllegroGraph**	**1.00**
	Cytoscape	0.67		**Neo4j**	**1.00**
	NetMiner	0.65		GraphX Apache Spark	0.92
	NetworkX	0.52		SparklingGraph	0.92
Information Fusion	**Grasphistry**	**1.00**	**Visualization**	**ORA-LITE/PRO**	**1.00**
	Netminer	0.89		**Grasphistry**	**1.00**
	Network Workbench	0.67		**Neo4j**	**1.00**
	ORA-LITE/PRO	0.67		Gephi	0.93
	Pajek	0.56		Cytoscape	0.93

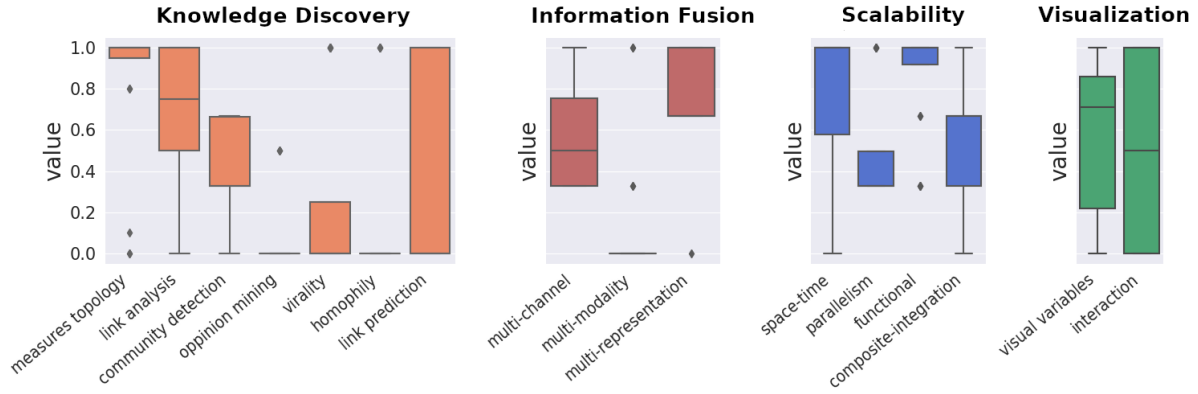


Figure 11: Un-aggregated dimension distributions for the top-5 SNA-software tools, each color represents a different dimension. The X-axis contains each of the features that form a dimension (SNA_{degree}), and the Y-axis their numerical value.

In order to further evaluate this phenomenon, the top 5 SNA-software tools for each dimension are shown in Table 4. Starting with the *Knowledge Discovery* dimension. The best performing SNA-software in this dimension is ORA-LITE/PRO and it scores better than the second-best tool, SNAP. This makes ORA-LITE/PRO an *outlier* regarding the *Knowledge Discovery* dimension (see Fig. 9). A similar case can be found for the *Information Fusion* dimension, where Graphistry tool provides the highest possible score (see Fig. 9). In this case Graphistry also scores better than the second-best tool, Netminer. Contrary, the *Scalability* dimension shows three equally good tools as top-performing (Graphistry, AllegroGraph and Neo4j), followed by GraphX Apache Spark and SparklingGraph. Finally, a similar case can be found on the *Visualization* dimension with the top tools (ORA-LITE/PRO, Graphistry and Neo4j) followed by Gephi and Cytoscape.

The analysis so far has shown that, the ORA-LITE/PRO and Graphistry tools, stand out from the rest in the *Knowledge Discovery* and *Information Fusion* dimensions. Below, the characteristics (or features) that have made these tools to stand out will be analyzed. To do so, each dimension has been split into its basic features (see Table 1, and sections from 6.1 to 6.4). Fig. 11 shows the distribution of each of the features that forms a dimension. Notice that in the *Knowledge Discovery* dimension, the *Opinion Mining* and the *Homophily* features are composed mostly by tools that have achieved a score of 0, and only a few of them have been able to achieve a higher score. Something similar happens with the *Virality* feature but in a less acute way. The tools that work with those features are the ones that appear in the top 5. Moreover, ORA-LITE/PRO is the only tool that has achieved a score (greater than 0) in every feature of the *Knowledge Discovery* dimension.

A similar case can be found when we analyze the *Information Fusion* dimension. In the *Multi-Modality* feature most of the cases are 0, and only a few are able of scoring something in this feature. In fact, Graphistry is the only tool that has scored a maximum rating in all of the features of the *Information Fusion* dimension. In general, a similar case can be found on the *Scalability* dimension. However, contrary to the *multi-modality* feature, several tools have achieved the maximum value in this category and not only one. Actually, all the tools that appear in the top 5 in that dimension have

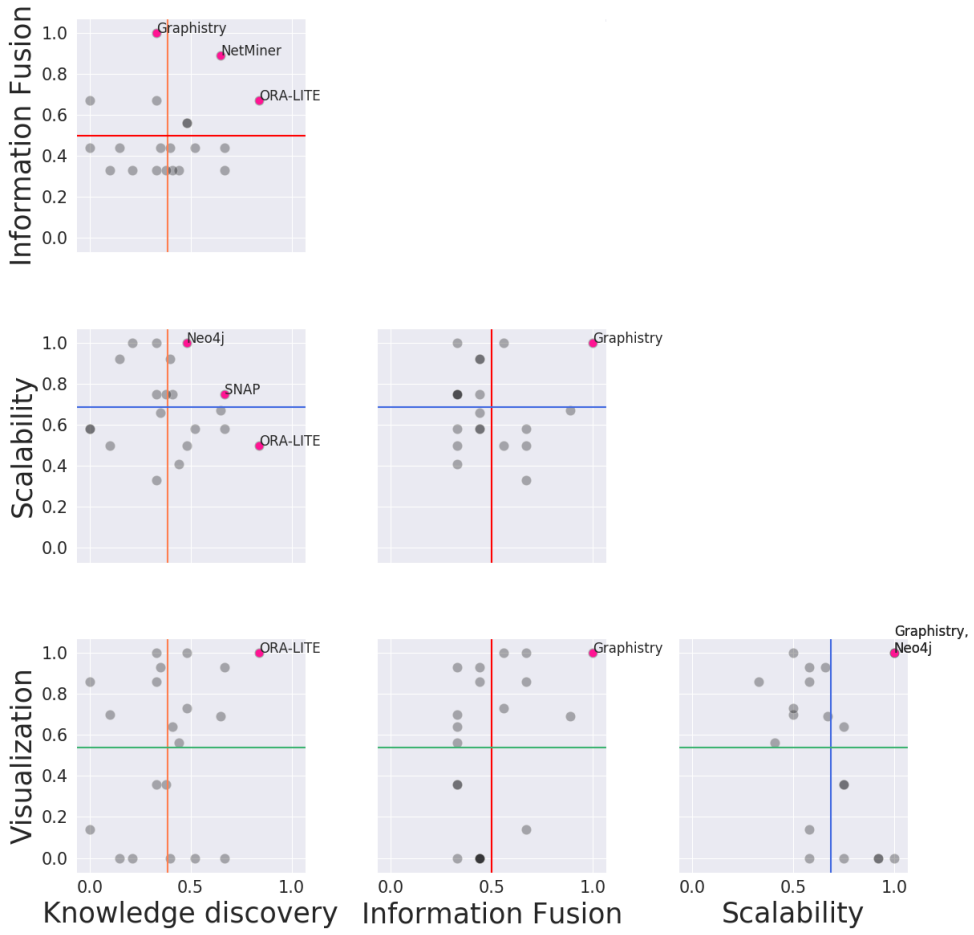


Figure 12: Pairing plot graphics, of the dimensions proposed for SNA-software. Each cell contains a graph where two different dimensions are depicted on the X and Y axis, and each point in the graph represents a tool. The horizontal and vertical lines show the average value at each dimension. The darker a point, the more tools have scored those values. Finally, the pink dots represent the Pareto front of a set, the members of the Pareto front are labeled with their names.

achieved a maximum score on that feature. Finally, the *visualization* dimension is the most homogeneous. Nevertheless, the *Visual Variables* feature scores slightly higher than the *Interaction* ones.

To sum up, we have noticed an uneven distribution on the 4 dimensions features. There are some features were nearly all tools score good, while in others only a few are able to obtain some scoring. This makes those tools stand out over the others. For example, the *Measures Topology*, *Link analysis* or *Funcional* features have high values for nearly all the tools analyzed, specially the *Measures Topology* one. Contrary, features like *Opinion Mining*, *Homophily* or *Multi-Modality* are tackled by very few tools. These last features can be used as a foundation of the guidelines that the next iteration of SNA tools must follow.

Up to now, the 4 dimensions have been analyzed independently. However, it is possible that exist different correlations, or relations, between the dimensions. For example, a proportional relationship between *Visualization* and *Knowledge Discovery* might be observed. It can be hypothesized that the more knowledge a tool can extract, the more visualization capabilities should it have (in order to process it). A similar hypothesis could be drawn between the *Visualization* and the *Information Fusion* dimensions following the same logic. Furthermore, a proportional relation could be also expected between the *Information Fusion* and the *Knowledge Discovery* dimensions. Since, the more types of information a tool is able to handle, the broader the type of possible analysis will be, and therefore the further the knowledge extraction will get. Contrary, an inversely proportional relationship is to be expected between the *Scalability* and the *Knowledge Discovery* dimensions. Fancier knowledge extraction techniques, usually require from a high amount of computational

power. Implementing these methods in a scalable way is a daunting task that is currently being tackled by the scientific community. Finally, the same logic could be applied between the *Scalability* and the *Information Fusion* dimensions.

Therefore, and in order to explore these relations, we have represented the (pair to pair) relations between all of the dimensions for the SNA-software tools analyzed. The results are shown in Fig. 12, this figure plots two different dimensions in the X and Y axis, and uses dots to represents the analyzed tools. The intensity of the dots indicates the number of tools that have scored a particular tuple of values. A pink dot indicates that a particular tools is a member of the Pareto Front of a graph. The members of the Pareto Front have been labeled with its tool name. Finally, to help the evaluation of the tool distributions, two colored vertical/horizontal lines have been added. These lines indicate the mean obtained for each particular dimension. Two types of graph are present in the matrix. On the one hand, there are the graphs where one tool dominates all the others. You can see this effect in *Scalability vs. Information Fusion*, *Visualization vs. Knowledge Discovery*, *Visualization vs. Information Fusion*, and the *Visualization vs. Scalability* graphs. On the other hand, there are the graphs where that dominance is not present. Notice the *Information Fusion vs. Knowledge Discovery* and the *Scalability vs. Knowledge Discovery* graphs.

Regarding the hypothesized relations, the data shows that the proportional relation between *Visualization* and *Knowledge Discovery*, and *Visualization* and *Information Fusion* is present in the data. Notice how in their respective graphs there is a higher condensation of points in the top right corner of the plot and how the number of dots grow from left to right and from top to bottom The points that have a zero value in the *Visualization* dimension have been ignored as those tools do not implement any kind of visualization. However, the hypothesized proportional relation between *Information Fusion* and *Knowledge Discovery* is not present in the data. Notice how the several fronts presented in the graph decreases from left to right. This has surprised the authors and could be interpreted as a future research niche. The data, a priory, shows that collecting extra information types and sources are not being fully exploited by the knowledge extraction algorithms. Concerning the hypothesized inversely proportional relations between the *Scalability* and the *Knowledge Discovery* dimensions , and the *Scalability* and the *Information Fusion* dimensions, the data have shown that both are present. Notice that the dominance fronts of both graphs decrease from left to right. This effect is far more acute between the *Scalability* and the *Knowledge Discovery* dimensions. Nevertheless, the *Scalability/Information Fusion* graph also shows a particular pattern not expected by the authors. The far dominance of *Graphistry* of every other tool in that graph is surprising. In fact, there are only two tools on the top right sector of the graph including *Graphistry*, and the other one is near the mean line. In the authors opinion this huge gap is another research niche, especially taking into account that *Graphistry* is a commercial tool and no other open-source one is near to its capabilities.

8 Conclusion, Future Trends and Challenges

In the last years, we have witnessed the increasing relevance of OSNs in our daily life. This popularity is produced by the high number of users that participate in these social media platforms every day and this participation results in huge amount of data generated by user interactions. OSN popularity and its exponential growth have led to an enormous interest in the analysis of this type of networks. There is a plethora of issues that can be studied from social media data, like the interconnections that originate the network, the structure, the evolution of the network, the identification of user communities, how the information flow and how this information is disseminated, or the patterns that can be extracted from them, just to mention a few of them. As a consequence of becoming a hot research area, the number of papers, conferences, journals, algorithms and tools has risen exponentially. This growth makes almost impossible to analyze in detail the current state of the art related to SNA. In order to limit the state of the art, and to present relevant review of the different research papers published in this topic, we have performed a scientometric study to define the most relevant research areas. From this analysis several fundamental research (graph theory and network analytics, community detection algorithms, information diffusion models, text mining and topic extraction, opinion mining and sentiment analysis), and application domains (health, marketing, tourism and hospitality, cyber-crime and cyber-terrorism, politics, detection of fake news and misinformation, and finally multimedia), have been studied in Sections 3 and 4.

Any analysis of OSNs rely on the representation of the data as a graph, and then some algorithms are applied to extract some valuable knowledge. In this sense, we have reviewed the basics of SNA techniques and algorithms in Section 3. The different algorithms are categorized around what kind of information is used in the analysis. In this sense, any researcher can extract information from the resulting graph, performing a structural-based analysis, or he/she can perform the analysis based on the content published in the OSN (content-based analysis). On the one hand, using the network structure any researcher can apply algorithms to detect the communities that compose the graph, or to study how the information is propagated through the network. On the other hand, using the content of the information published the standard analysis techniques rely on solving the problems of topic extraction, opinion mining and sentiment analysis. Finally, we have also analyzed the current state of the art regarding multimedia content; i.e., how the different media (text, audio and video) is used to perform SNA.

In spite of the previous state of the art analysis, and due to the size of the research community around SNA, it is quite difficult for a novel researcher, or someone (even experienced) who wants to start his/her research on this area, to select the most appropriate tool or algorithm. In order to offer a starting point to the community, we have proposed in this work four research questions that any researcher should answer, or at least keep in mind, before starting his/her research in this area. From these research questions it have been proposed several dimensions, as a tool to obtain a quantitative assessment of the current maturity of SNA technologies, allowing both to better understand what are the main strengths and weaknesses of these technologies, and to look for future trends and possible improvements in the next years in this area.

To perform this quantitative assessment some specific metrics, or *degrees*, have been defined in Section 6. And finally, a quantitative evaluation of a set of 20 popular SNA-software tools have been carried out, to show how these dimensions (and their related metrics) can be used to evaluate these technologies, in Section 7.

The main conclusions from the research questions proposed, their related dimensions and metrics proposed, and the assessment of the SNA- software tools carried out, can be briefly summarized as follows:

1. **What can I discover?**. This question is related to the different types of knowledge that the tool is able to extract. The goal of this question is to quantify the capacity of the tool to extract valuable knowledge from the data. To answer this question, the dimension *Pattern & Knowledge discovery* (and its related $d_{Value}(t)$ metric) has been considered. From the current study, it can be concluded that some kind of analyzes, like topology measures, link analysis or static community detection, are fairly common in the tools analyzed. While other analyzes, like dynamic community detection, opinion mining, virality or homophily, are quite rare. In our opinion, this phenomenon is linked to the lack of proportional relation between the *Information Fusion* and *Knowledge Discovery*. This suggests that the content of an OSN is not being fully exploited by actual tools.
2. **What is the limit?**. Answering this question the researcher will understand the scalability of the tool. This question is quite important due to the amount of data that can be extracted from OSNs. This question has been addressed through the *Scalability* dimension (and its related $d_{Volume}(t)$ metric). As the quantitative analysis of the tools shown in Section 7, it is clear that most of the analyzed tools are capable of handling fairly big graphs (around 100.000 nodes), they are very customizable (their code is publicly available), and allow communication with other tools via an API, even thought just a few achieves fully integration with other applications. However, few tools are capable of doing BigData and the ones that can, have a low/medium average Knowledge Discovery capabilities. Taking into account the fast growth of the OSNs, the size of handled networks (to millions or hundred of millions nodes and vertex) will be swift increased in this tools, jointly to other capabilities, as the fusion and integration from different sources, and with different tools, to improve the knowledge discovery reached by these tools.
3. **What kind of data can I integrate?**. This relevant question, related to the capacity to integrate and fusion information from current SNA technologies, has been analyzed through the *Information Fusion & Integration* dimension (and its related $d_{Variety}(t)$ metric). Again, the quantitative analysis of the tools carried out, shows that most of the analyzed tools used complex graph representation (multilayered graphs or hypergraphs), are capable of processing a medium amount of different data types (two or three different types, and are only capable of extracting data from one unique OSN. Therefore, and related to this dimension, it can be expected a very significant increase in research related to the fusion and integration of information using different types of data formats, and when possible, from different OSNs.
4. **What can I show?**. Finally, this last question was explored using the *Visualization* dimension (and its related $d_{Visual}(t)$ metric), and how it has been shown, although there exist a large number of information visualization, and tools that provide flexible methods to visualize the information, this is still an open problem in the area. From the quantitative analysis related to this dimension, it can be concluded that the visualization capabilities of the tools where more evenly distributed compared to the rest of the dimensions. However, we have observed a lack of tools with high Scalability and Visualization capabilities. Taking the absolutely necessity to provide visualization tools to the end users and practitioners, the research and improvements in this area will be a high (and hot) topic in SNA in the next years, for example, in areas as dynamic community finding, data analytics or pattern finding to mention but a few.

Finally, we need to make a reflection on the dimensions and metrics proposed. What is proposed here is an initial work, derived from an intense dedication to the area of SNA in the last ten years. These dimensions, and the defined metrics (or degrees), cannot (and should not) be considered as the only ones that can be defined, even the definition cannot be considered as complete. From the analysis of the state of the art, we have selected those more relevant (from our perspective) features that could be used to better identify and reflect the state of these technologies. It is quite probable,

that some highly relevant characteristics have not been considered by authors. On the other hand, the fact that several tools such as Grasphistry have a value of 1.0 in the dimensions of Information Fusion, Scalability, or Visualization, or that other tools such as Neo4J, also reach the value of 1.0 in dimensions such as Scalability or Visualization, do not mean that these features cannot be improved in the future. These values only indicate that given the current state of technology (and using our evaluation rubric), a higher value is achieved than other systems or tools for those dimensions. Obviously, and given the huge and fast growth of these technologies, we are sure that these values will change in the coming years, but the authors think that these dimensions, along with their related metrics, could be an important decision tool for future researchers, and practitioners, in the field of SNA.

Appendix: Open Access to Social Network Analysis Dimensions

In order to allow researchers from social sciences, science and engineering, SNA analysts and professionals, developers and engineers, etc. not only to access the data used in this article, but to foster for a future collaboration among the community interested in network analysis, a website has been designed to facilitate accessing to:

- An open github repository to allow including new software, documents, open papers, and technical data, related to SNA technology.
- *Rubric.odt*: The evaluation rubric designed to assess the SNA-software tools.
- *analized_tools.ods*: The specific evaluation made for the SNA-software tools carried out in this paper.
- *README.md*: A collaborative website to evaluate SNA-software (tools, frameworks, algorithms, etc.) by the community, included as part of the github repository.
- *4Dimensions.pdf*: A brief summary of this paper.

The SNA 4-Dimensions website:

https://ai-da-sna.github.io/

We would like to encourage the community to provide its own evaluations, of both the tools that have been evaluated in this paper as well as others in which you have previous experience. This collaboration will promote the use of SNA technologies and fostering new developments, but it will not be possible without the cooperation of the community, so your contribution will be highly appreciated.

Acknowledgements

This work has been supported by several research grants: Spanish Ministry of Science and Education under TIN2014-56494-C4-4-P grant (DeepBio), European Union, under ISFP-POLICE ACTION: 823701-ISFP-2017-AG-RAD grant (YoungRes), and Comunidad Autónoma de Madrid under S2018/TCS-4566 grant (CYNAMON). We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan V GPU used for this research.

References

[1] Danah M Boyd and Nicole B Ellison. Social network sites: Definition, history, and scholarship. *Journal of computer-mediated Communication*, 13(1):210–230, 2007.

[2] Alfred Reginald Radcliffe-Brown. The social organization of australian tribes. *Oceania*, 1(1):34–63, 1930.

[3] Alfred Reginald Radcliffe-Brown. On social structure. *The Journal of the Royal Anthropological Institute of Great Britain and Ireland*, 70(1):1–12, 1940.

[4] John Arundel Barnes. Class and committees in a norwegian island parish. *Human relations*, 7(1):39–58, 1954.

[5] John Scott. Social network analysis. *Sociology*, 22(1):109–127, 1988.

[6] Charu C Aggarwal. An introduction to social network data analytics. In *Social network data analytics*, pages 1–15. Springer, 2011.

[7] Xindong Wu, Xingquan Zhu, Gong-Qing Wu, and Wei Ding. Data mining with big data. *IEEE transactions on knowledge and data engineering*, 26(1):97–107, 2013.

[8] Peng Wang, BaoWen Xu, YuRong Wu, and XiaoYu Zhou. Link prediction in social networks: the state-of-the-art. *Science China Information Sciences*, 58(1):1–38, 2015.

[9] Stephen P Borgatti, Martin G Everett, and Jeffrey C Johnson. *Analyzing social networks*. Sage, 2018.

[10] Romualdo Pastor-Satorras, Claudio Castellano, Piet Van Mieghem, and Alessandro Vespignani. Epidemic processes in complex networks. *Reviews of modern physics*, 87(3):925, 2015.

[11] Dave Chaffey. Global social media research summary 2019, 2019.

[12] Adrien Guille, Hakim Hacid, Cecile Favre, and Djamel A. Zighed. Information diffusion in online social networks: A survey. *SIGMOND Record*, 42(2):17–28, 2013.

[13] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 990–998. ACM, 2008.

[14] Doug Laney. 3d data management: Controlling data volume, velocity and variety. *META group research note*, 6(70):1, 2001.

[15] Josep M Pujol, Vijay Erramilli, Georgos Siganos, Xiaoyuan Yang, Nikos Laoutaris, Parminder Chhabra, and Pablo Rodriguez. The little engine (s) that could: scaling online social networks. *ACM SIGCOMM Computer Communication Review*, 41(4):375–386, 2011.

[16] Ken Wakita and Toshiyuki Tsurumi. Finding community structure in mega-scale social networks. In *Proceedings of the 16th international conference on World Wide Web*, pages 1275–1276. ACM, 2007.

[17] Gema Bello-Orgaz, Jason J Jung, and David Camacho. Social big data: Recent achievements and new challenges. *Information Fusion*, 28:45–59, 2016.

[18] Soujanya Poria, Erik Cambria, Newton Howard, Guang-Bin Huang, and Amir Hussain. Fusing audio, visual and textual clues for sentiment analysis from multimodal content. *Neurocomputing*, 174:50–59, 2016.

[19] Aaron Beach, Mike Gartrell, Xinyu Xing, Richard Han, Qin Lv, Shivakant Mishra, and Karim Seada. Fusing mobile, sensor, and social data to fully enable context-aware computing. In *Proceedings of the eleventh workshop on mobile computing systems & applications*, pages 60–65. ACM, 2010.

[20] Linton C Freeman. Visualizing social networks. *Journal of social structure*, 1(1):4, 2000.

[21] Ulrik Brandes and Dorothea Wagner. Analysis and visualization of social networks. In *Graph drawing software*, pages 321–340. Springer, 2004.

[22] Rafael Ball. *An Introduction to Bibliometrics: New Development and Trends*. Chandos Publishing, 2017.

[23] John McLevey and Reid McIlroy-Young. Introducing metaknowledge: Software for computational research in information science, network analysis, and science of science. *Journal of Informetrics*, 11(1):176–197, 2017.

[24] Werner Marx, Lutz Bornmann, Andreas Barth, and Loet Leydesdorff. Detecting the historical roots of research fields by reference publication year spectroscopy (rpys). *Journal of the Association for Information Science and Technology*, 65(4):751–764, 2014.

[25] Jordan A Comins and Thomas W Hussey. Compressing multiple scales of impact detection by reference publication year spectroscopy. *Journal of Informetrics*, 9(3):449–454, 2015.

[26] Stanley Milgram. The small world problem. *Psychology today*, 2(1):60–67, 1967.

[27] Mark S Granovetter. The strength of weak ties. In *Social networks*, pages 347–367. Elsevier, 1977.

[28] Linton C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 40(1):35–41, 1977.

[29] Linton C Freeman. Centrality in social networks conceptual clarification. *Social networks*, 1(3):215–239, 1978.

[30] James S Coleman. Social capital in the creation of human capital. *American journal of sociology*, 94:S95–S120, 1988.

[31] Johan Bollen, Huina Mao, and Xiaojun Zeng. Twitter mood predicts the stock market. *Journal of Computational Science*, 2(1):1 – 8, 2011.

[32] Thomas W. Valente. Network interventions. *Science*, 337(6090):49–53, 2012.

[33] Miller McPherson, Lynn Smith-Lovin, and James M Cook. Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, 27(1):415–444, 2001.

[34] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[35] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, June 1998.

[36] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, 2002.

[37] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, oct 2008.

[38] Mark EJ Newman. Fast algorithm for detecting community structure in networks. *Physical review E*, 69(6):066133, 2004.

[39] Mark EJ Newman and Michelle Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

[40] Aaron Clauset, Mark EJ Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical review E*, 70(6):066111, 2004.

[41] Aaron Clauset. Finding local community structure in networks. *Phys. Rev. E*, 72:026132, August 2005.

[42] Gergely Palla, Imre Derényi, Illés Farkas, and Tamás Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *nature*, 435(7043):814, 2005.

[43] Gergely Palla, Albert-László Barabási, and Tamás Vicsek. Quantifying social group evolution. *Nature*, 446(7136):664, 2007.

[44] Pascal Pons and Matthieu Latapy. Computing communities in large networks using random walks. In pInar Yolum, Tunga Güngör, Fikret Gürgen, and Can Özturan, editors, *Computer and Information Sciences - ISCIS 2005*, pages 284–293, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[45] Pascal Pons and Matthieu Latapy. Computing communities in large networks using random walks. *J. Graph Algorithms Appl.*, 10(2):191–218, 2006.

[46] Martin Rosvall and Carl T Bergstrom. An information-theoretic framework for resolving community structure in complex networks. *Proceedings of the National Academy of Sciences*, 104(18):7327–7331, 2007.

[47] Martin Rosvall and Carl T Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 105(4):1118–1123, 2008.

[48] Jussi M Kumpula, Mikko Kivelä, Kimmo Kaski, and Jari Saramäki. Sequential algorithm for fast clique percolation. *Physical Review E*, 78(2):026109, 2008.

[49] Randolf Rotta and Andreas Noack. Multilevel local search algorithms for modularity clustering. *Journal of Experimental Algorithmics (JEA)*, 16:2–3, 2011.

[50] Sandro Cavallari, Erik Cambria, Hongyun Cai, Kevin Chang, and Vincent Zheng. Embedding both finite and infinite communities on graph. *IEEE Computational Intelligence Magazine*, 14(3):39–50, 2019.

[51] Stanley Wasserman and Katherine Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994.

[52] Steven E Stemler. Content analysis. *Emerging trends in the social and behavioral sciences: An Interdisciplinary, Searchable, and Linkable Resource*, pages 1–14, 2015.

[53] John A Barnes and Frank Harary. Graph theory in network analysis. *Social networks*, 5(2):235–244, 1983.

[54] Stephen P Borgatti, Ajay Mehra, Daniel J Brass, and Giuseppe Labianca. Network analysis in the social sciences. *science*, 323(5916):892–895, 2009.

[55] Dean Lusher, Johan Koskinen, and Garry Robins. *Exponential random graph models for social networks: Theory, methods, and applications*. Cambridge University Press, 2013.

[56] Peter J Carrington, John Scott, and Stanley Wasserman. *Models and methods in social network analysis*, volume 28. Cambridge university press, 2005.

[57] Fernando Vega-Redondo. *Complex social networks*. Number 44. Cambridge University Press, 2007.

[58] My T Thai and Panos M Pardalos. *Handbook of optimization in complex networks: theory and applications*, volume 57. Springer science & business media, 2011.

[59] Douglas Brent West. *Introduction to graph theory*, volume 2. Prentice hall Upper Saddle River, NJ, 1996.

[60] John Adrian Bondy and Uppaluri Siva Ramachandra Murty. *Graph theory with applications*, volume 290. Citeseer, 1976.

[61] Gema Bello-Orgaz, Julio Hernandez-Castro, and David Camacho. Detecting discussion communities on vaccination in twitter. *Future Generation Computer Systems*, 66:125–136, 2017.

[62] Linton Freeman. The development of social network analysis. *A Study in the Sociology of Science*, 1, 2004.

[63] Raúl Lara-Cabrera, Antonio González Pardo, Karim Benouaret, Noura Faci, Djamal Benslimane, and David Camacho. Measuring the radicalisation risk in social networks. *IEEE Access*, 5:10892–10900, 2017.

[64] Nathalie Henry, Jean-Daniel Fekete, and Michael J McGuffin. Nodetrix: a hybrid visualization of social networks. *IEEE transactions on visualization and computer graphics*, 13(6):1302–1309, 2007.

[65] Mark S Handcock, David R Hunter, Carter T Butts, Steven M Goodreau, and Martina Morris. statnet: Software tools for the representation, visualization, analysis and simulation of network data. *Journal of statistical software*, 24(1):1548, 2008.

[66] Barry Wellman. Network analysis: Some basic principles. *Sociological theory*, 1(1):155–200, 1983.

[67] Bing Liu and Lei Zhang. A survey of opinion mining and sentiment analysis. In *Mining text data*, pages 415–463. Springer, 2012.

[68] Amanda L Traud, Peter J Mucha, and Mason A Porter. Social structure of facebook networks. *Physica A: Statistical Mechanics and its Applications*, 391(16):4165–4180, 2012.

[69] Tarik Crnovrsanin, Chris W Muelder, Robert Faris, Diane Felmlee, and Kwan-Liu Ma. Visualization techniques for categorical analysis of social networks with multiple edge sets. *Social Networks*, 37:56–64, 2014.

[70] Jure Leskovec, Daniel Huttenlocher, and Jon Kleinberg. Signed networks in social media. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 1361–1370. ACM, 2010.

[71] Thomas Zaslavsky. Signed graphs. *Discrete Applied Mathematics*, 4(1):47–74, 1982.

[72] Haifa Gmati, Amira Mouakher, Antonio Gonzalez-Pardo, and David Camacho. A new algorithm for communities detection in social networks with node attributes. *Journal of Ambient Intelligence and Humanized Computing (Special Issue on Computational Intelligence for Social Mining)*, pages 1–13, 2018.

[73] Jonathan L Gross and Jay Yellen. *Handbook of graph theory*. CRC press, 2004.

[74] Mark EJ Newman and Juyong Park. Why social networks are different from other types of networks. *Physical review E*, 68(3):036122, 2003.

[75] Béla Bollobás and Bollobás Béla. *Random graphs*. Number 73. Cambridge university press, 2001.

[76] Mark EJ Newman, Duncan J Watts, and Steven H Strogatz. Random graph models of social networks. *Proceedings of the National Academy of Sciences*, 99(suppl 1):2566–2572, 2002.

[77] Paul Erdős and Alfréd Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, 5(1):17–60, 1960.

[78] Edgar N Gilbert. Random graphs. *The Annals of Mathematical Statistics*, 30(4):1141–1144, 1959.

[79] Réka Albert and Albert-László Barabási. Statistical mechanics of complex networks. *Reviews of modern physics*, 74(1):47, 2002.

[80] J-P Onnela, Jari Saramäki, Jorkki Hyvönen, György Szabó, David Lazer, Kimmo Kaski, János Kertész, and A-L Barabási. Structure and tie strengths in mobile communication networks. *Proceedings of the national academy of sciences*, 104(18):7332–7336, 2007.

[81] Albert-László Barabási, Réka Albert, and Hawoong Jeong. Mean-field theory for scale-free random networks. *Physica A: Statistical Mechanics and its Applications*, 272(1-2):173–187, 1999.

[82] Geir Agnarsson and Raymond Greenlaw. *Graph theory: Modeling, applications, and algorithms*. Pearson/Prentice Hall, 2007.

[83] Fred Buckley and Marty Lewinter. *A friendly introduction to graph theory*. Prentice Hall New Jersey, 2003.

[84] Reza Zafarani, Mohammad Ali Abbasi, and Huan Liu. *Social media mining: an introduction*. Cambridge University Press, 2014.

[85] Diane J Cook and Lawrence B Holder. *Mining graph data*. John Wiley & Sons, 2006.

[86] Shimon Even. *Graph algorithms*. Cambridge University Press, 2011.

[87] Jierui Xie, Stephen Kelley, and Boleslaw K Szymanski. Overlapping community detection in networks: The state-of-the-art and comparative study. *ACM Computing Surveys (CSUR)*, 45(4):43, 2013.

[88] Santo Fortunato. Community detection in graphs. *Physics reports*, 486(3-5):75–174, 2010.

[89] Gema Bello-Orgaz, Héctor D Menéndez, and David Camacho. Adaptive k-means algorithm for overlapped graph clustering. *International journal of neural systems*, 22(05):1250018, 2012.

[90] Antonio Gonzalez-Pardo and David Camacho. Design of japanese tree frog algorithm for community finding problems. In *Intelligent Data Engineering and Automated Learning – IDEAL 2018*, pages 307–315. Springer International Publishing, 2018.

[91] Ravi Kannan, Santosh Vempala, and Adrian Vetta. On clusterings: Good, bad and spectral. *Journal of the ACM (JACM)*, 51(3):497–515, 2004.

[92] Immanuel M. Bomze, Marco Budinich, Panos M. Pardalos, and Marcello Pelillo. The maximum clique problem. In *Handbook of Combinatorial Optimization*, pages 1–74. Kluwer Academic Publishers, 1999.

[93] Mark EJ Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582, 2006.

[94] Thomas Richardson, Peter J Mucha, and Mason A Porter. Spectral tripartitioning of networks. *Physical Review E*, 80(3):036111, 2009.

[95] Gaoxia Wang, Yi Shen, and Ming Ouyang. A vector partitioning approach to detecting community structure in complex networks. *Computers & Mathematics with Applications*, 55(12):2746–2752, 2008.

[96] Haijun Zhou and Reinhard Lipowsky. Network brownian motion: A new method to measure vertex-vertex proximity and to identify communities and subcommunities. In *Computational Science-ICCS 2004*, pages 1062–1069. Springer, 2004.

[97] Yihong Dong, Yueting Zhuang, Ken Chen, and Xiaoying Tai. A hierarchical clustering algorithm based on fuzzy graph connectedness. *Fuzzy Sets and Systems*, 157(13):1760–1774, 2006.

[98] Gema Bello-Orgaz and David Camacho. Evolutionary clustering algorithm for community detection using graph-based information. In *2014 IEEE Congress on Evolutionary Computation (CEC)*, pages 930–937. IEEE, 2014.

[99] Gema Bello-Orgaz, Sancho Salcedo-Sanz, and David Camacho. A multi-objective genetic algorithm for overlapping community detection based on edge encoding. *Information Sciences*, 462:290–314, 2018.

[100] Lei Tang and Huan Liu. Graph mining applications to social network analysis. In *Managing and Mining Graph Data*, pages 487–513. Springer, 2010.

[101] James Abello, Mauricio GC Resende, and Sandra Sudarsky. Massive quasi-clique detection. In *Latin American symposium on theoretical informatics*, pages 598–612. Springer, 2002.

[102] Ravi Kumar, Jasmine Novak, and Andrew Tomkins. Structure and evolution of online social networks. In *Link mining: models, algorithms, and applications*, pages 337–357. Springer, 2010.

[103] Balabhaskar Balasundaram, Sergiy Butenko, and Illya V Hicks. Clique relaxations in social network analysis: The maximum k-plex problem. *Operations Research*, 59(1):133–142, 2011.

[104] Sergey N Dorogovtsev, Alexander V Goltsev, and Jose Ferreira F Mendes. K-core organization of complex networks. *Physical review letters*, 96(4):040601, 2006.

[105] Gary William Flake, Steve Lawrence, C Lee Giles, et al. Efficient identification of web communities. In *KDD*, volume 2000, pages 150–160, 2000.

[106] Peter D Hoff, Adrian E Raftery, and Mark S Handcock. Latent space approaches to social network analysis. *Journal of the american Statistical association*, 97(460):1090–1098, 2002.

[107] Mark S Handcock, Adrian E Raftery, and Jeremy M Tantrum. Model-based clustering for social networks. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 170(2):301–354, 2007.

[108] Fionn Murtagh and Pedro Contreras. Algorithms for hierarchical clustering: an overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(1):86–97, 2012.

[109] Katherine A Heller and Zoubin Ghahramani. Bayesian hierarchical clustering. In *Proceedings of the 22nd international conference on Machine learning*, pages 297–304. ACM, 2005.

[110] Giulio Rossetti and Rémy Cazabet. Community discovery in dynamic networks: a survey. *ACM Computing Surveys (CSUR)*, 51(2):35, 2018.

[111] Rémy Cazabet and Frédéric Amblard. *Dynamic Community Detection*, pages 404–414. Springer New York, New York, NY, 2014.

[112] Rémy Cazabet, Hideaki Takeda, Masahiro Hamasaki, and Frédéric Amblard. Using dynamic community detection to identify trends in user-generated content. *Social Network Analysis and Mining*, 2(4):361–371, 2012.

[113] Sitaram Asur, Srinivasan Parthasarathy, and Duygu Ucar. An event-based framework for characterizing the evolutionary behavior of interaction graphs. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 3(4):16, 2009.

[114] Derek Greene, Donal Doyle, and Padraig Cunningham. Tracking the evolution of communities in dynamic social networks. In *2010 international conference on advances in social networks analysis and mining*, pages 176–183. IEEE, 2010.

[115] Piotr Bródka, Stanisław Saganowski, and Przemysław Kazienko. Ged: the method for group evolution discovery in social networks. *Social Network Analysis and Mining*, 3(1):1–14, 2013.

[116] Etienne Gael Tajeuna, Mohamed Bouguessa, and Shengrui Wang. Tracking the evolution of community structures in time-evolving social networks. In *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 1–10. IEEE, 2015.

[117] Zhixiao Wang, Zechao Li, Guan Yuan, Yunlian Sun, Xiaobin Rui, and Xinguang Xiang. Tracking the evolution of overlapping communities in dynamic social networks. *Knowledge-Based Systems*, 157:81–97, 2018.

[118] Etienne Gael Tajeuna, Mohamed Bouguessa, and Shengrui Wang. Modeling and predicting community structure changes in time-evolving social networks. *IEEE Transactions on Knowledge and Data Engineering*, 31(6):1166–1180, 2018.

[119] Deepayan Chakrabarti, Ravi Kumar, and Andrew Tomkins. Evolutionary clustering. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 554–560. ACM, 2006.

[120] Angel Panizo-LLedot, Gema Bello-Orgaz, and David Camacho. A multi-objective genetic algorithm for detecting dynamic communities using a local search driven immigrant's scheme. *Future Generation Computer Systems. Available online 15 November 2019.*, pages 1–16, 2019.

[121] A Panizo, G Bello-Orgaz, and David Camacho. A genetic algorithm with local search based on label propagation for detecting dynamic communities. In *International Symposium on Intelligent and Distributed Computing*, pages 319–328. Springer, 2018.

[122] Eneko Osaba, Javier Del Ser, Angel Panizo, David Camacho, Akemi Galvez, and Andres Iglesias. Combining bio-inspired meta-heuristics and novelty search for community detection over evolving graph streams. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pages 1329–1335. ACM, 2019.

[123] Francesco Folino and Clara Pizzuti. Multiobjective evolutionary community detection for dynamic networks. In *GECCO*, pages 535–536, 2010.

[124] Yu-Ru Lin, Yun Chi, Shenghuo Zhu, Hari Sundaram, and Belle L Tseng. Analyzing communities and their evolutions in dynamic social networks. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 3(2):8, 2009.

[125] Yu-Ru Lin, Jimeng Sun, Hari Sundaram, Aisling Kelliher, Paul Castro, and Ravi Konuru. Community discovery via metagraph factorization. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 5(3):17, 2011.

[126] Lei Tang, Huan Liu, Jianping Zhang, and Zohreh Nazeri. Community evolution in dynamic multi-mode networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 677–685. ACM, 2008.

[127] Robert Görke, Pascal Maillard, Andrea Schumm, Christian Staudt, and Dorothea Wagner. Dynamic graph clustering combining modularity and smoothness. *Journal of Experimental Algorithmics (JEA)*, 18:1–5, 2013.

[128] Hamidreza Alvari, Alireza Hajibagheri, Gita Sukthankar, and Kiran Lakkaraju. Identifying community structures in dynamic networks. *Social Network Analysis and Mining*, 6(1):77, 2016.

[129] Xu Zhou, Xiaohui Zhao, and Yanheng Liu. A multiobjective discrete bat algorithm for community detection in dynamic networks. *Applied Intelligence*, 48(9):3081–3093, 2018.

[130] Chayant Tantipathananandh, Tanya Berger-Wolf, and David Kempe. A framework for community identification in dynamic social networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 717–726. ACM, 2007.

[131] Tianbao Yang, Yun Chi, Shenghuo Zhu, Yihong Gong, and Rong Jin. Detecting communities and their evolutions in dynamic social networks—a bayesian approach. *Machine learning*, 82(2):157–189, 2011.

[132] Peter J Mucha, Thomas Richardson, Kevin Macon, Mason A Porter, and Jukka-Pekka Onnela. Community structure in time-dependent, multiscale, and multiplex networks. *science*, 328(5980):876–878, 2010.

[133] Bivas Mitra, Lionel Tabourier, and Camille Roth. Intrinsically dynamic network communities. *Computer Networks*, 56(3):1041–1053, 2012.

[134] Kevin S Xu and Alfred O Hero. Dynamic stochastic blockmodels for time-evolving social networks. *IEEE Journal of Selected Topics in Signal Processing*, 8(4):552–562, 2014.

[135] Fuchen Liu, David Choi, Lu Xie, and Kathryn Roeder. Global spectral clustering in dynamic networks. *Proceedings of the National Academy of Sciences*, 115(5):927–932, 2018.

[136] Jimeng Sun, Christos Faloutsos, Christos Faloutsos, Spiros Papadimitriou, and Philip S Yu. Graphscope: parameter-free mining of large time-evolving graphs. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 687–696. ACM, 2007.

[137] Jeffrey Chan, James Bailey, and Christopher Leckie. Discovering correlated spatio-temporal changes in evolving graphs. *Knowledge and Information Systems*, 16(1):53–96, 2008.

[138] Dongsheng Duan, Yuhua Li, Yanan Jin, and Zhengding Lu. Community mining on dynamic weighted directed graphs. In *Proceedings of the 1st ACM international workshop on Complex networks meet information & knowledge management*, pages 11–18. ACM, 2009.

[139] Thomas Aynaud and Jean-Loup Guillaume. Multi-step community detection and hierarchical time segmentation in evolving networks. In *Proceedings of the 5th SNA-KDD workshop*, 2011.

[140] Leto Peel and Aaron Clauset. Detecting change points in the large-scale structure of evolving networks. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[141] Yuriy Hulovatyy and Tijana Milenković. Scout: simultaneous time segmentation and community detection in dynamic networks. *Scientific reports*, 6:37557, 2016.

[142] Remy Cazabet, Frederic Amblard, and Chihab Hanachi. Detection of overlapping communities in dynamical social networks. In *2010 IEEE second international conference on social computing*, pages 309–314. IEEE, 2010.

[143] Nam P Nguyen, Thang N Dinh, Ying Xuan, and My T Thai. Adaptive algorithms for detecting community structure in dynamic social networks. In *2011 Proceedings IEEE INFOCOM*, pages 2282–2290. IEEE, 2011.

[144] Jingyong Li, Lan Huang, Tian Bai, Zhe Wang, and Hongsheng Chen. Cdbia: a dynamic community detection method based on incremental analysis. In *2012 International Conference on Systems and Informatics (ICSAI2012)*, pages 2224–2228. IEEE, 2012.

[145] Anita Zakrzewska and David A Bader. A dynamic algorithm for local community detection in graphs. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*, pages 559–564. ACM, 2015.

[146] Giulio Rossetti, Luca Pappalardo, Dino Pedreschi, and Fosca Giannotti. Tiles: an online algorithm for community discovery in dynamic social networks. *Machine Learning*, 106(8):1213–1241, 2017.

[147] Souâad Boudebza, Rémy Cazabet, Faiçal Azouaou, and Omar Nouali. Olcpm: An online framework for detecting overlapping communities in dynamic social networks. *Computer Communications*, 123:36–51, 2018.

[148] Tanja Falkowski, Anja Barth, and Myra Spiliopoulou. Studying community dynamics with an incremental graph mining algorithm. *AMCIS 2008 Proceedings*, page 29, 2008.

[149] Sajid Yousuf Bhat and Muhammad Abulaish. Octracker: A density-based framework for tracking the evolution of overlapping communities in osns. In *Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2012)*, pages 501–505. IEEE Computer Society, 2012.

[150] Tiphaine Viard, Matthieu Latapy, and Clémence Magnien. Computing maximal cliques in link streams. *Theoretical Computer Science*, 609:245–252, 2016.

[151] Anne-Sophie Himmel, Hendrik Molter, Rolf Niedermeier, and Manuel Sorge. Enumerating maximal cliques in temporal graphs. In *Proceedings of the 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 337–344. IEEE Press, 2016.

[152] Rong-Hua Li, Jiao Su, Lu Qin, Jeffrey Xu Yu, and Qiangqiang Dai. Persistent community search in temporal networks. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, pages 797–808. IEEE, 2018.

[153] Tiphaine Viard, Clémence Magnien, and Matthieu Latapy. Enumerating maximal cliques in link streams with durations. *Information Processing Letters*, 133:44–48, 2018.

[154] Suman Banerjee and Bithika Pal. On the enumeration of maximal (δ, γ)-cliques of a temporal network. In *Proceedings of the ACM India Joint International Conference on Data Science and Management of Data*, pages 112–120. ACM, 2019.

[155] Hongchao Qin, Rong-Hua Li, Guoren Wang, Lu Qin, Yurong Cheng, and Ye Yuan. Mining periodic cliques in temporal networks. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, pages 1130–1141. IEEE, 2019.

[156] Raj Agnihotri, Rebecca Dingus, Michael Y Hu, and Michael T Krush. Social media: Influencing customer satisfaction in b2b sales. *Industrial Marketing Management*, 53:172–180, 2016.

[157] Mingli Zhang, Lingyun Guo, Mu Hu, and Wenhua Liu. Influence of customer engagement with company social networks on stickiness: Mediating effect of customer value creation. *International Journal of Information Management*, 37(3):229–240, 2017.

[158] Rosemary Thackeray, Brad L Neiger, Carl L Hanson, and James F McKenzie. Enhancing promotional strategies within social marketing programs: use of web 2.0 social media. *Health promotion practice*, 9(4):338–343, 2008.

[159] Christy Ashley and Tracy Tuten. Creative strategies in social media marketing: An exploratory study of branded social content and consumer engagement. *Psychology & Marketing*, 32(1):15–27, 2015.

[160] Philip N Howard and Bence Kollanyi. Bots,# strongerin, and# brexit: computational propaganda during the uk-eu referendum. *Available at SSRN 2798311*, 2016.

[161] Gunn Enli. Twitter as arena for the authentic outsider: Exploring the social media campaigns of trump and clinton in the 2016 us presidential election. *European journal of communication*, 32(1):50–61, 2017.

[162] Ańgel Panizo-LLedot, Javier Torregrosa, Gema Bello-Orgaz, Joshua Thorburn, and David Camacho. Describing alt-right communities and their discourse on twitter during the 2018 us mid-term elections. In *International Conference on Complex Networks and Their Applications*, pages 427–439. Springer, 2019.

[163] Jytte Klausen. Tweeting the jihad: Social media networks of western foreign fighters in syria and iraq. *Studies in Conflict & Terrorism*, 38(1):1–22, 2015.

[164] Jytte Klausen, Eliane Tschaen Barbieri, Aaron Reichlin-Melnick, and Aaron Y Zelin. The youtube jihadists: A social network analysis of al-muhajiroun's propaganda campaign. *Perspectives on Terrorism*, 6(1), 2012.

[165] Raúl Lara-Cabrera, Antonio Gonzalez-Pardo, and David Camacho. Statistical analysis of risk assessment factors and metrics to evaluate radicalisation in twitter. *Future Generation Computer Systems*, 2017.

[166] Hunt Allcott and Matthew Gentzkow. Social media and fake news in the 2016 election. *Journal of economic perspectives*, 31(2):211–36, 2017.

[167] Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. Fake news detection on social media: A data mining perspective. *ACM SIGKDD Explorations Newsletter*, 19(1):22–36, 2017.

[168] Adrien Guille, Hakim Hacid, Cecile Favre, and Djamel A Zighed. Information diffusion in online social networks: A survey. *ACM Sigmod Record*, 42(2):17–28, 2013.

[169] Mark EJ Newman. The structure and function of complex networks. *SIAM review*, 45(2):167–256, 2003.

[170] Giovanna Miritello, Esteban Moro, and Rubén Lara. Dynamical strength of social ties in information spreading. *Physical Review E*, 83(4):045102, 2011.

[171] Chuang Liu and Zi-Ke Zhang. Information spreading on dynamic social networks. *Communications in Nonlinear Science and Numerical Simulation*, 19(4):896–904, 2014.

[172] Romualdo Pastor-Satorras and Alessandro Vespignani. Epidemic spreading in scale-free networks. *Physical review letters*, 86(14):3200, 2001.

[173] David Kempe, Jon Kleinberg, and Éva Tardos. Influential nodes in a diffusion model for social networks. In *International Colloquium on Automata, Languages, and Programming*, pages 1127–1138. Springer, 2005.

[174] Jacob Goldenberg, Barak Libai, and Eitan Muller. Talk of the network: A complex systems look at the underlying process of word-of-mouth. *Marketing letters*, 12(3):211–223, 2001.

[175] Jacob Goldenberg, Barak Libai, and Eitan Muller. Using complex systems analysis to advance marketing theory development: Modeling heterogeneity effects on new product growth through stochastic cellular automata. *Academy of Marketing Science Review*, 9(3):1–18, 2001.

[176] Duncan J Watts. A simple model of global cascades on random networks. *Proceedings of the National Academy of Sciences*, 99(9):5766–5771, 2002.

[177] Erik Cambria, Haixun Wang, and Bebo White. Guest editorial: Big social data analysis. *Knowledge-Based Systems*, 69:1–2, 2014.

[178] Meeyoung Cha, Haewoon Kwak, Pablo Rodriguez, Yong-Yeol Ahn, and Sue Moon. I tube, you tube, everybody tubes: analyzing the world's largest user generated content video system. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 1–14. ACM, 2007.

[179] Chung-Yi Weng, Wei-Ta Chu, and Ja-Ling Wu. Rolenet: Movie analysis from the perspective of social networks. *IEEE Transactions on Multimedia*, 11(2):256–271, 2009.

[180] Amir Gandomi and Murtaza Haider. Beyond the hype: Big data concepts, methods, and analytics. *International journal of information management*, 35(2):137–144, 2015.

[181] Xindong Wu, Xingquan Zhu, Gong-Qing Wu, and Wei Ding. Data mining with big data. *IEEE transactions on knowledge and data engineering*, 26(1):97–107, 2014.

[182] Lev Manovich. Trending: the promises and the challenges of big social data. *Debates in the digital humanities*, pages 460–475, 2011.

[183] Zeynep Tufekci. Big questions for social media big data: Representativeness, validity and other methodological pitfalls. In *Eighth International AAAI Conference on Weblogs and Social Media*, 2014.

[184] C Pal and Andrew McCallum. CC prediction with grapical models. In *Conference on Email and Anti-Spam*, 2006.

[185] O Bar-Yossef, I Guy, R Lempel, Y Maarek, and V Soroka. Cluster ranking with an application to mining mailbox networks. *Knowledge and Information Systems*, 14(1):101–139, 2008.

[186] M Roth, A Ben-David, D Deutscher, G Flysher, I Horn, A Leichtberg, N Leiser, Y Matias, and R. Merom. Suggesting friends using the implicit social graph. In *KDD*, 2010.

[187] M De Choudhury, W Mason, J Hofman, and D Watts. Inferring relevant social networks from interpersonal communication. In *WWW*, 2010.

[188] Yichuan Cai and Yi Chen. Mass: a multi-facet domainspecific influential blogger mining system. In *International Conference on Data Engineering*, pages 1109–1112, 2010.

[189] Chang Sun, Bing-quan Liu, Cheng-jie Sun, De-Yuan Zhang, and Xiaolong Wang. Simrank: A link analysis based blogger recommendation algorithm using text similarity. In *International Conference on Machine Learning and Cybernetics*, pages 3368–3373, 2010.

[190] Leonidas Akritidis, Dimitrios Katsaros, and Panayiotis Bozanis. Identifying the productive and influential bloggers in a community. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 41(5):759–764, 2011.

[191] Eunyoung Moon and Sangki Han. A qualitative method to find influencers using similarity-based approach in the blogosphere. *International Journal of Social Computing and Cyber-Physical Systems*, 1(1):56–78, 2011.

[192] Praphul Chandra, Erik Cambria, and Amir Hussain. Clustering social networks using interaction semantics and sentics. In J Wang, G Yen, and M Polycarpou, editors, *Advances in Neural Networks*, volume 7367 of *Lecture Notes in Computer Science*, pages 379–385. Springer, 2012.

[193] Dheeraj Rajagopal, Daniel Olsher, Erik Cambria, and Kenneth Kwok. Commonsense-based topic modeling. In *ACM KDD*, 2013.

[194] Guangyuan Piao and John Breslin. Inferring user interests in microblogging social networks: A survey. *User Modeling and User-Adapted Interaction*, 28(3):277–329, 2018.

[195] Andrew McCallum, X Wang, and A Corrada-Emmanuel. Topic and role discovery in social networks withexperiments on enron and academic email. *Journal of Artificial Intelligence Research*, 30:249–272, 2007.

[196] N Pathak, C Delong, A Banerjee, and K Erickson. Social topic models for community extraction. In *SNA-KDD Workshop*, 2008.

[197] T Wang, H Liu, J He, and X Du. Mining user interests from information sharing behaviors in social media. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 85–98, 2013.

[198] J Wang, W Zhao, Y He, and X Li. Infer user interests via link structure regularization. *ACM Transactions on Intelligent Systems and Technology*, 5(2):23, 2014.

[199] J Kang and H Lee. Modeling user interest in social media using news media and wikipedia. *Information Systems*, 65:52–64, 2017.

[200] C Jipmo, G Quercini, and N Bennacer. Frisk: A multilingual approach to find twitter interests via wikipedia. In *International Conference on Advanced Data Mining and Applications*, pages 243–256, 2017.

[201] S Faralli, G Stilo, and P Velardi. Automatic acquisition of a taxonomy of microblogs users' interests. *Web Semantics: Science, Services and Agents on the World Wide Web*, 45:23–40, 2017.

[202] F Zarrinkalam, M Kahani, and E Bagheri. Mining user interests over active topics on social networks. *Information Processing and Management*, 54(2):339–357, 2018.

[203] A Trikha, F Zarrinkalam, and E Bagheri. Topic-association mining for user interest detection. In *European Conference on Information Retrieval*, pages 665–671, 2018.

[204] Erik Cambria, Newton Howard, Yunqing Xia, and Tat-Seng Chua. Computational intelligence for big social data analysis. *IEEE Computational Intelligence Magazine*, 11(3):8–9, 2016.

[205] Erik Cambria, Soujanya Poria, Federica Bisio, Rajiv Bajpai, and Iti Chaturvedi. The CLSA model: A novel framework for concept-level sentiment analysis. In *LNCS*, volume 9042, pages 3–22. Springer, 2015.

[206] Yash Mehta, Navonil Majumder, Alexander Gelbukh, and Erik Cambria. Recent trends in deep learning based personality detection. *Artificial Intelligence Review*, 2020.

[207] A Bandhakavi, N Wiratunga, S Massie, and P Deepak. Lexicon generation for emotion analysis from text. *IEEE Intelligent Systems*, 32(1):102–108, 2017.

[208] Feng Xu, Jianfei Yu, and Rui Xia. Instance-based domain adaptation via multi-clustering logistic approximation. *IEEE Intelligent Systems*, 33(1):78–88, 2018.

[209] Qinjuan Yang, Yanghui Rao, Haoran Xie, Jiahai Wang, Fu Lee Wang, and Wai Hong Chan. Segment-level joint topic-sentiment model for online review analysis. *IEEE Intelligent Systems*, 34(1):43–50, 2019.

[210] Albert Weichselbraun, Stefan Gindl, Fabian Fischer, Svitlana Vakulenko, and Arno Scharl. Aspect-based extraction and analysis of affective knowledge from social media streams. *IEEE Intelligent Systems*, 32(3):80–88, 2017.

[211] Iti Chaturvedi, Ranjan Satapathy, Sandro Cavallari, and Erik Cambria. Fuzzy commonsense reasoning for multimodal sentiment analysis. *Pattern Recognition Letters*, 125(264-270), 2019.

[212] Rada Mihalcea and Aparna Garimella. What men say, what women hear: Finding gender-specific meaning shades. *IEEE Intelligent Systems*, 31(4):62–67, 2016.

[213] Abeer Bukeer, Giorgio Roffo, and Alessandro Vinciarelli. Type like a man! inferring gender from keystroke dynamics in live-chats. *IEEE Intelligent Systems*, 34(6):53–59, 2019.

[214] Debanjan Mahata, Jasper Friedrichs, Rajiv Ratn Shah, and Jing Jiang. Detecting personal intake of medicine from twitter. *IEEE Intelligent Systems*, 33(4):87–95, 2018.

[215] Syed Arbaaz Qureshi, Sriparna Saha, Mohammed Hasanuzzaman, and Gael Dias. Multitask representation learning for multimodal estimation of depression level. *IEEE Intelligent Systems*, 34(5):45–52, 2019.

[216] Monireh Ebrahimi, Amir Hossein, and Amit Sheth. Challenges of sentiment analysis for dynamic events. *IEEE Intelligent Systems*, 32(5):70–75, 2017.

[217] Ana Valdivia, Victoria Luzon, and Francisco Herrera. Sentiment analysis in tripadvisor. *IEEE Intelligent Systems*, 32(4):72–77, 2017.

[218] Md Shad Akhtar, Asif Ekbal, Sunny Narayan, and Vikram Singh. No, that never happened!! investigating rumors on twitter. *IEEE Intelligent Systems*, 33(5):8–15, 2018.

[219] Julio Reis, Andre Correia, Fabricio Murai, Adriano Veloso, and Fabricio Benevenuto. Supervised learning for fake news detection. *IEEE Intelligent Systems*, 34(2):76–81, 2019.

[220] Charles Welch, Veronica Perez-Rosas, Jonathan Kummerfeld, and Rada Mihalcea. Learning from personal longitudinal dialog data. *IEEE Intelligent Systems*, 34(4):16–23, 2019.

[221] Nishith Pathak, Sandeep Mane, Jaideep Srivastava, and Noshir Contractor. Knowledge perception analysis in a social network. In *SIAM International Conference on Data Mining*, 2006.

[222] A Bermingham, Mike Conway, L McInerney, N O'Hare, and A Smeaton. Combining social network analysis and sentiment analysis to explore the potential for online radicalisation. In *IEEE International Conference on Advances in Social Network Analysis and Mining*, pages 231–236, 2009.

[223] Wojciech Gryc and Karo Moilanen. Leveraging textual sentiment analysis with social network modeling: Sentiment analysis of political blogs in the 2008 u.s. presidential election. In *From Text to Political Positions Workshop*, 2010.

[224] Mohammadreza Shams, Mohammadtaghi Saffar, Azadeh Shakery, and Heshaam Faili. Applying sentiment and social network analysis in user modeling. In *CICLing*, pages 526–539, 2012.

[225] JA Morente-Molinera, G Kou, K Samuylov, R Urena, and E Herrera-Viedma. Carrying out consensual group decision making processes under social networks using sentiment analysis over comparative expressions. *Knowledge-Based Systems*, 165:335–345, 2019.

[226] Kirsten P Smith and Nicholas A Christakis. Social networks and health. *Annu. Rev. Sociol*, 34:405–429, 2008.

[227] Duncan Chambers, Paul Wilson, Carl Thompson, and Melissa Harden. Social network analysis in healthcare settings: a systematic scoping review. *PloS one*, 7(8):e41911, 2012.

[228] Lisa F. Berkman, Thomas Glass, Ian Brissette, and Teresa E. Seeman. From social integration to health: Durkheim in the new millennium. *Social Science and Medicine*, 51(6):843 – 857, 2000.

[229] Sheldon Cohen and Denise Janicki-Deverts. Can we improve our physical health by altering our social networks? *Perspectives on Psychological Science*, 4(4):375–378, 2009.

[230] Peggy A Thoits. Mechanisms linking social ties and support to physical and mental health. *Journal of health and social behavior*, 52(2):145–161, 2011.

[231] Debra Umberson and Jennifer Karas Montez. Social relationships and health: A flashpoint for health policy. *Journal of health and social behavior*, 51(1_suppl):S54–S66, 2010.

[232] Javier Andreu-Perez, Carmen CY Poon, Robert D Merrifield, Stephen TC Wong, and Guang-Zhong Yang. Big data for health. *IEEE journal of biomedical and health informatics*, 19(4):1193–1208, 2015.

[233] Nicholas A Christakis and James H Fowler. The spread of obesity in a large social network over 32 years. *New England journal of medicine*, 357(4):370–379, 2007.

[234] US Department of Health, Human Services, et al. Preventing tobacco use among youth and young adults: a report of the surgeon general, 2012.

[235] Elisa M Trucco, Craig R Colder, and William F Wieczorek. Vulnerability to peer influence: A moderated mediation study of early adolescent alcohol use initiation. *Addictive behaviors*, 36(7):729–736, 2011.

[236] Sonia Livingstone. Taking risky opportunities in youthful content creation: teenagers' use of social networking sites for intimacy, privacy and self-expression. *New media & society*, 10(3):393–411, 2008.

[237] Becky Freeman. New media and tobacco control. *Tobacco control*, 21(2):139–144, 2012.

[238] Grace C. Huang, Jennifer B. Unger, Daniel Soto, Kayo Fujimoto, Mary Ann Pentz, Maryalice Jordan-Marsh, and Thomas W. Valente. Peer influences: The impact of online and offline friendship networks on adolescent smoking and alcohol use. *Journal of Adolescent Health*, 54(5):508 – 514, 2014.

[239] Julien I.E. Hoffman. Chapter 27 - linear regression. In Julien I.E. Hoffman, editor, *Biostatistics for Medical and Biomedical Practitioners*, pages 451 – 500. Academic Press, 2015.

[240] Nicholas A Christakis and James H Fowler. The collective dynamics of smoking in a large social network. *New England journal of medicine*, 358(21):2249–2258, 2008.

[241] Mark E Larsen, Tjeerd W Boonstra, Philip J Batterham, Bridianne O'Dea, Cecile Paris, and Helen Christensen. We feel: mapping emotion on twitter. *IEEE journal of biomedical and health informatics*, 19(4):1246–1252, 2015.

[242] C Paquet, D Coulombier, R Kaiser, and M Ciotti. Epidemic intelligence: a new framework for strengthening disease surveillance in europe. *Euro surveillance: bulletin europeen sur les maladies transmissibles European communicable disease bulletin*, 11(12):212–214, 2005.

[243] Gema Bello-Orgaz, Julio Hernandez-Castro, and David Camacho. A survey of social web mining applications for disease outbreak detection. In *Intelligent Distributed Computing VIII*, pages 345–356. Springer, 2015.

[244] Aron Culotta. Towards detecting influenza epidemics by analyzing twitter messages. In *Proceedings of the first workshop on social media analytics*, pages 115–122. ACM, 2010.

[245] Eiji Aramaki, Sachiko Maskawa, and Mizuki Morita. Twitter catches the flu: detecting influenza epidemics using twitter. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1568–1576. Association for Computational Linguistics, 2011.

[246] Todd Bodnar and Marcel Salathé. Validating models for disease detection using twitter. In *Proceedings of the 22nd international conference on World Wide Web companion*, pages 699–702. International World Wide Web Conferences Steering Committee, 2013.

[247] Paul Harrigan, Uwana Evers, Morgan Miles, and Timothy Daly. Customer engagement with tourism social media brands. *Tourism Management*, 59:597–609, 2017.

[248] Chiranjeev Kohli, Rajneesh Suri, and Anuj Kapoor. Will social media kill branding? *Business Horizons*, 58(1):35–44, 2015.

[249] W Glynn Mangold and David J Faulds. Social media: The new hybrid element of the promotion mix. *Business horizons*, 52(4):357–365, 2009.

[250] Lisette De Vries, Sonja Gensler, and Peter SH Leeflang. Popularity of brand posts on brand fan pages: An investigation of the effects of social media marketing. *Journal of interactive marketing*, 26(2):83–91, 2012.

[251] Jeryl Whitelock, John W Cadogan, Shintaro Okazaki, and Charles R Taylor. Social media and international advertising: theoretical challenges and future directions. *International marketing review*, 2013.

[252] Kunal Swani, George R Milne, Brian P Brown, A George Assaf, and Naveen Donthu. What messages to post? evaluating the popularity of social media communications in business versus consumer markets. *Industrial Marketing Management*, 62:77–87, 2017.

[253] Rodney Graeme Duffett. Facebook advertising's influence on intention-to-purchase and purchase amongst millennials. *Internet Research*, 25(4):498–526, 2015.

[254] François A. Carrillat, Alain d'Astous, and Emilie Morissette Grégoire. Leveraging social media to enhance recruitment effectiveness: A facebook experiment. *Internet Research*, 24(4):474–495, 2014.

[255] Simon Hudson, Li Huang, Martin S Roth, and Thomas J Madden. The influence of social media interactions on consumer–brand relationships: A three-country study of brand perceptions and marketing behaviors. *International Journal of Research in Marketing*, 33(1):27–41, 2016.

[256] Giampaolo Viglia, Roberta Minazzi, and Dimitrios Buhalis. The influence of e-word-of-mouth on hotel occupancy rate. *International Journal of Contemporary Hospitality Management*, 28(9):2035–2051, 2016.

[257] Eunice Kim, Yongjun Sung, and Hamsu Kang. Brand followers' retweeting behavior on twitter: How brand relationships influence brand electronic word-of-mouth. *Computers in Human Behavior*, 37:18–25, 2014.

[258] Shu-Chuan Chu and Yoojung Kim. Determinants of consumer engagement in electronic word-of-mouth (ewom) in social networking sites. *International journal of Advertising*, 30(1):47–75, 2011.

[259] Bernard J Jansen, Mimi Zhang, Kate Sobel, and Abdur Chowdury. Twitter power: Tweets as electronic word of mouth. *Journal of the American society for information science and technology*, 60(11):2169–2188, 2009.

[260] Sitaram Asur, Bernardo Huberman, et al. Predicting the future with social media. In *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM International Conference on*, volume 1, pages 492–499. IEEE, 2010.

[261] Mya Pronschinske, Mark D Groza, and Matthew Walker. Attracting facebook'fans': the importance of authenticity and engagement as a social networking strategy for professional sport teams. *Sport marketing quarterly*, 21(4):221, 2012.

[262] Jeff McCarthy, Jennifer Rowley, Catherine Jane Ashworth, and Elke Pioch. Managing brand presence through social media: the case of uk football clubs. *Internet Research*, 24(2):181–204, 2014.

[263] Keith S Coulter and Anne Roggeveen. "like it or not" consumer responses to word-of-mouth communication in on-line social networks. *Management Research Review*, 35(9):878–899, 2012.

[264] Edward C Malthouse, Michael Haenlein, Bernd Skiera, Egbert Wege, and Michael Zhang. Managing customer relationships in the social media era: Introducing the social crm house. *Journal of Interactive Marketing*, 27(4):270–280, 2013.

[265] Kevin J Trainor, James Mick Andzulis, Adam Rapp, and Raj Agnihotri. Social media technology usage and customer relationship performance: A capabilities-based examination of social crm. *Journal of Business Research*, 67(6):1201–1208, 2014.

[266] Jesse N Moore, Christopher D Hopkins, and Mary Anne Raymond. Utilization of relationship-oriented social media in the selling process: a comparison of consumer (b2c) and industrial (b2b) salespeople. *Journal of Internet Commerce*, 12(1):48–75, 2013.

[267] Hélia Gonçalves Pereira, Maria de Fátima Salgueiro, and Inês Mateus. Say yes to facebook and get your customers involved! relationships in a world of social networks. *Business Horizons*, 57(6):695–702, 2014.

[268] Ana María Munar and Jens Kr Steen Jacobsen. Motivations for sharing tourism experiences through social media. *Tourism management*, 43:46–54, 2014.

[269] Jingjing Li, Lizhi Xu, Ling Tang, Shouyang Wang, and Ling Li. Big data in tourism research: A literature review. *Tourism Management*, 68:301–323, 2018.

[270] Marianna Sigala, Evangelos Christou, and Ulrike Gretzel. *Social media in travel, tourism and hospitality: Theory, practice and cases*. Ashgate Publishing, Ltd., 2012.

[271] Roberta Minazzi. Social media marketing in tourism and hospitality, 2015.

[272] Nathan Bennett, Raynald Harvey Lemelin, Rhonda Koster, and Isabel Budke. A capital assets framework for appraising and building capacity for tourism development in aboriginal protected area gateway communities. *Tourism Management*, 33(4):752–766, 2012.

[273] Daniel Leung, Rob Law, Hubert Van Hoof, and Dimitrios Buhalis. Social media in tourism and hospitality: A literature review. *Journal of travel & tourism marketing*, 30(1-2):3–22, 2013.

[274] Khim-Yong Goh, Cheng-Suang Heng, and Zhijie Lin. Social media brand community and consumer behavior: Quantifying the relative impact of user-and marketer-generated content. *Information Systems Research*, 24(1):88–107, 2013.

[275] Alan Wilson, Hilary Murphy, and Jesus Cambra Fierro. Hospitality and travel: The nature and implications of user-generated content. *Cornell hospitality quarterly*, 53(3):220–228, 2012.

[276] Xun Xu and Yibai Li. The antecedents of customer satisfaction and dissatisfaction toward various types of hotels: A text mining approach. *International journal of hospitality management*, 55:57–69, 2016.

[277] Yue Guo, Stuart J Barnes, and Qiong Jia. Mining meaning from online ratings and reviews: Tourist satisfaction analysis using latent dirichlet allocation. *Tourism Management*, 59:467–483, 2017.

[278] Zheng Xiang, Zvi Schwartz, John H Gerdes Jr, and Muzaffer Uysal. What can big data and text analytics tell us about hotel guest experience and satisfaction? *International Journal of Hospitality Management*, 44:120–130, 2015.

[279] Sun-Jae Doh and Jang-Sun Hwang. How consumers evaluate ewom (electronic word-of-mouth) messages. *CyberPsychology & Behavior*, 12(2):193–197, 2009.

[280] Soujanya Poria, Iti Chaturvedi, Erik Cambria, and Federica Bisio. Sentic LDA: Improving on LDA with semantic similarity for aspect-based sentiment analysis. In *IJCNN*, pages 4465–4473, 2016.

[281] Erik Cambria, Soujanya Poria, Devamanyu Hazarika, and Kenneth Kwok. SenticNet 5: Discovering conceptual primitives for sentiment analysis by means of context embeddings. In *AAAI*, pages 1795–1802, 2018.

[282] Claudia Guerreiro, Erik Cambria, and Hien Nguyen. Understanding the role of social media in backpacker tourism. In *ICDM*, pages 530–537, 2019.

[283] Ya-Han Hu, Yen-Liang Chen, and Hui-Ling Chou. Opinion mining from online hotel reviews–a text summarization approach. *Information Processing & Management*, 53(2):436–449, 2017.

[284] Gloria Bordogna, Luca Frigerio, Alfredo Cuzzocrea, and Giuseppe Psaila. Clustering geo-tagged tweets for advanced big data analytics. In *2016 IEEE International Congress on Big Data (BigData Congress)*, pages 42–51. IEEE, 2016.

[285] Huy Quan Vu, Gang Li, Rob Law, and Ben Haobin Ye. Exploring the travel behaviors of inbound tourists to hong kong using geotagged photos. *Tourism Management*, 46:222–232, 2015.

[286] Xin Lu, Changhu Wang, Jiang-Ming Yang, Yanwei Pang, and Lei Zhang. Photo2trip: generating travel routes from geo-tagged photos for trip planning. In *Proceedings of the 18th ACM international conference on Multimedia*, pages 143–152. ACM, 2010.

[287] Ning Deng and Xiang Robert Li. Feeling a destination through the "right" photos: A machine learning model for dmos' photo selection. *Tourism Management*, 65:267–278, 2018.

[288] Abdul Majid, Ling Chen, Gencai Chen, Hamid Turab Mirza, Ibrar Hussain, and John Woodward. A context-aware personalized travel recommendation system based on geotagged social media data mining. *International Journal of Geographical Information Science*, 27(4):662–684, 2013.

[289] Chih-Hao Ku and Gondy Leroy. A decision support system: Automated crime report analysis and classification for e-government. *Government Information Quarterly*, 31(4):534–544, 2014.

[290] Peter Phillips and Ickjai Lee. Mining co-distribution patterns for large crime datasets. *Expert Systems with Applications*, 39(14):11556–11563, 2012.

[291] Spencer Chainey, Lisa Tompson, and Sebastian Uhlig. The utility of hotspot mapping for predicting spatial patterns of crime. *Security Journal*, 21(1):4–28, 2008.

[292] Matthew S Gerber. Predicting crime using twitter and kernel density estimation. *Decision Support Systems*, 61:115–125, 2014.

[293] United Nations Office On Drugs and Crime. The use of the Internet for terrorist purposes. Technical report, Vienna, 2012.

[294] R Thompson. Radicalization and the use of social media. *Journal of Strategic Security*, 4(4):167–190, 2011.

[295] Katie Cohen, Fredrik Johansson, Lisa Kaati, and Jonas Clausen Mork. Detecting linguistic markers for radical violence in social media. *Terrorism and Political Violence*, 26(1):246–256, 2014.

[296] J Reid Meloy, Jens Hoffmann, Angela Guldimann, and David James. The role of warning behaviors in threat assessment: An exploration and suggested typology. *Behavioral sciences & the law*, 30(3):256–279, 2012.

[297] Javier Torregrosa, Joshua Thorburn, Raúl Lara-Cabrera, David Camacho, and Humberto M. Trujillo. Linguistic analysis of pro-isis users on twitter. *Behavioral Sciences of Terrorism and Political Aggression*, 0(0):1–15, 2019.

[298] Miriam Fernandez, Antonio Gonzalez-Pardo, and Harith Alani. Radicalisation influence in social media. *Semantic Web Journal*, pages In–Press, 2019.

[299] Mohammed Lalou, Mohammed Amin Tahraoui, and Hamamache Kheddouci. The critical node detection problem in networks: A survey. *Computer Science Review*, 28:92 – 117, 2018.

[300] R Chulaka Gunasekara, Kishan Mehrotra, and Chilukuri K Mohan. Multi-objective optimization to identify key players in large social networks. *Social Network Analysis and Mining*, 5(1):21, 2015.

[301] Optimizing network attacks by artificial bee colony. *Information Sciences*, 377:30 – 50, 2017.

[302] Sukhdeep Singh, Navrati Saxena, Abhishek Roy, and HanSeok Kim. A survey on 5g network technologies from social perspective. *IETE Technical Review*, 34(1):30–39, 2017.

[303] Giuseppe Araniti, Antonino Orsino, Leonardo Militano, Li Wang, and Antonio Iera. Context-aware information diffusion for alerting messages in 5g mobile social networks. *IEEE Internet of Things Journal*, 4(2):427–436, 2016.

[304] Luigi Atzori, Antonio Iera, Giacomo Morabito, and Michele Nitti. The social internet of things (siot)–when social networks meet the internet of things: Concept, architecture and network characterization. *Computer networks*, 56(16):3594–3608, 2012.

[305] Philip N Howard, Aiden Duffy, Deen Freelon, Muzammil M Hussain, Will Mari, and Marwa Maziad. Opening closed regimes: what was the role of social media during the arab spring? *Available at SSRN 2595096*, 2011.

[306] Shelley Boulianne. Social media use and participation: a meta-analysis of current research. *Information, Communication & Society*, 18(5):524–538, 2015.

[307] Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. Automated hate speech detection and the problem of offensive language. In *Eleventh international aaai conference on web and social media*, 2017.

[308] Anna Schmidt and Michael Wiegand. A survey on hate speech detection using natural language processing. In *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media*, pages 1–10, 2017.

[309] Michael D Conover, Jacob Ratkiewicz, Matthew Francisco, Bruno Gonçalves, Filippo Menczer, and Alessandro Flammini. Political polarization on twitter. In *Fifth international AAAI conference on weblogs and social media*, 2011.

[310] Symeon Papadopoulos, Yiannis Kompatsiaris, Athena Vakali, and Ploutarchos Spyridonos. Community detection in social media. *Data Mining and Knowledge Discovery*, 24(3):515–554, 2012.

[311] Eytan Bakshy, Itamar Rosenn, Cameron Marlow, and Lada Adamic. The role of social networks in information diffusion. In *Proceedings of the 21st international conference on World Wide Web*, pages 519–528. ACM, 2012.

[312] Karsten Müller and Carlo Schwarz. Fanning the flames of hate: Social media and hate crime. *Available at SSRN 3082972*, 2018.

[313] Sylvia Jaki and Tom De Smedt. Right-wing german hate speech on twitter: Analysis and automatic detection. *Manuscript submitted*, 2018.

[314] Pete Burnap and Matthew L Williams. Cyber hate speech on twitter: An application of machine classification and statistical modeling for policy and decision making. *Policy & Internet*, 7(2):223–242, 2015.

[315] Antoine Boutet, Hyoungshick Kim, and Eiko Yoneki. What's in twitter, i know what parties are popular and who you are supporting now! *Social Network Analysis and Mining*, 3(4):1379–1391, 2013.

[316] Anatoliy Gruzd and Jeffrey Roy. Investigating political polarization on twitter: A canadian perspective. *Policy & Internet*, 6(1):28–45, 2014.

[317] Anjie Fang, Iadh Ounis, Philip Habel, Craig Macdonald, and Nut Limsopatham. Topic-centric classification of twitter user's political orientation. In *Proceedings of the 38th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 791–794. ACM, 2015.

[318] Javier Borge-Holthoefer, Walid Magdy, Kareem Darwish, and Ingmar Weber. Content and network dynamics behind egyptian political polarization on twitter. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, pages 700–711. ACM, 2015.

[319] Mert Ozer, Nyunsu Kim, and Hasan Davulcu. Community detection in political twitter networks using nonnegative matrix factorization methods. In *Proceedings of the 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, ASONAM '16, pages 81–88. IEEE Press, 2016.

[320] Michael D Conover, Bruno Gonçalves, Jacob Ratkiewicz, Alessandro Flammini, and Filippo Menczer. Predicting the political alignment of twitter users. In *2011 IEEE third international conference on privacy, security, risk and trust and 2011 IEEE third international conference on social computing*, pages 192–199. IEEE, 2011.

[321] Elanor Colleoni, Alessandro Rozza, and Adam Arvidsson. Echo chamber or public sphere? predicting political orientation and measuring political homophily in twitter using big data. *Journal of communication*, 64(2):317–332, 2014.

[322] Stefan Stieglitz and Linh Dang-Xuan. Political communication and influence through microblogging–an empirical analysis of sentiment in twitter messages and retweet behavior. In *2012 45th Hawaii International Conference on System Sciences*, pages 3500–3509. IEEE, 2012.

[323] Pablo Barberá, John T Jost, Jonathan Nagler, Joshua A Tucker, and Richard Bonneau. Tweeting from left to right: Is online political communication more than an echo chamber? *Psychological science*, 26(10):1531–1542, 2015.

[324] Edson C Tandoc Jr, Zheng Wei Lim, and Richard Ling. Defining "fake news" a typology of scholarly definitions. *Digital journalism*, 6(2):137–153, 2018.

[325] Soroush Vosoughi, Deb Roy, and Sinan Aral. The spread of true and false news online. *Science*, 359(6380):1146–1151, 2018.

[326] Vian Bakir and Andrew McStay. Fake news and the economy of emotions: Problems, causes, solutions. *Digital journalism*, 6(2):154–175, 2018.

[327] Niall J Conroy, Victoria L Rubin, and Yimin Chen. Automatic deception detection: Methods for finding fake news. *Proceedings of the Association for Information Science and Technology*, 52(1):1–4, 2015.

[328] David MJ Lazer, Matthew A Baum, Yochai Benkler, Adam J Berinsky, Kelly M Greenhill, Filippo Menczer, Miriam J Metzger, Brendan Nyhan, Gordon Pennycook, David Rothschild, et al. The science of fake news. *Science*, 359(6380):1094–1096, 2018.

[329] Johannes Fürnkranz. A study using n-gram features for text categorization. *Austrian Research Institute for Artifical Intelligence*, 3(1998):1–10, 1998.

[330] Martin Potthast, Johannes Kiesel, Kevin Reinartz, Janek Bevendorff, and Benno Stein. A stylometric inquiry into hyperpartisan and fake news. *arXiv preprint arXiv:1702.05638*, 2017.

[331] Zhiwei Jin, Juan Cao, Yongdong Zhang, and Jiebo Luo. News verification by exploiting conflicting social viewpoints in microblogs. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[332] Eugenio Tacchini, Gabriele Ballarin, Marco L Della Vedova, Stefano Moret, and Luca de Alfaro. Some like it hoax: Automated fake news detection in social networks. *arXiv preprint arXiv:1704.07506*, 2017.

[333] Natali Ruchansky, Sungyong Seo, and Yan Liu. Csi: A hybrid deep model for fake news detection. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 797–806. ACM, 2017.

[334] Sejeong Kwon, Meeyoung Cha, Kyomin Jung, Wei Chen, and Yajun Wang. Prominent features of rumor propagation in online social media. In *2013 IEEE 13th International Conference on Data Mining*, pages 1103–1108. IEEE, 2013.

[335] Rongrong Ji, Donglin Cao, Yiyi Zhou, and Fuhai Chen. Survey of visual sentiment prediction for social media analysis. *Frontiers of Computer Science*, 10(4):602–611, 2016.

[336] Qimin Cheng, Qian Zhang, Peng Fu, Conghuan Tu, and Sen Li. A survey and analysis on automatic image annotation. *Pattern Recognition*, 79:242–259, 2018.

[337] Seema Wazarkar and Bettahally N Keshavamurthy. A survey on image data analysis through clustering techniques for real world applications. *Journal of Visual Communication and Image Representation*, 55:596–626, 2018.

[338] Aditya Khosla, Atish Das Sarma, and Raffay Hamid. What makes an image popular? In *Proceedings of the 23rd international conference on World wide web*, pages 867–876. ACM, 2014.

[339] Francesco Gelli, Tiberio Uricchio, Marco Bertini, Alberto Del Bimbo, and Shih-Fu Chang. Image popularity prediction in social media using sentiment and context features. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 907–910. ACM, 2015.

[340] Michele Merler, Liangliang Cao, and John R Smith. You are what you tweet... pic! gender prediction based on semantic analysis of social media images. In *2015 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2015.

[341] Kasthuri Jayarajah and Archan Misra. Can instagram posts help characterize urban micro-events? In *2016 19th International Conference on Information Fusion (FUSION)*, pages 130–137. IEEE, 2016.

[342] Miriam Redi, Daniele Quercia, Lindsay Graham, and Samuel Gosling. Like partying? your face says it all. predicting the ambiance of places with profile pictures. In *Ninth International AAAI Conference on Web and Social Media*, 2015.

[343] Yuheng Hu, Lydia Manikonda, and Subbarao Kambhampati. What we instagram: A first analysis of instagram photo content and user types. In *Eighth International AAAI conference on weblogs and social media*, 2014.

[344] Leqi Liu, Daniel Preotiuc-Pietro, Zahra Riahi Samani, Mohsen E Moghaddam, and Lyle Ungar. Analyzing personality through social media profile picture choice. In *Tenth international AAAI conference on web and social media*, 2016.

[345] Fabio Celli, Elia Bruni, and Bruno Lepri. Automatic personality and interaction style recognition from facebook profile pictures. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 1101–1104. ACM, 2014.

[346] Xingjie Wei and David Stillwell. How smart does your profile image look?: Estimating intelligence from social network profile images. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 33–40. ACM, 2017.

[347] Patricia A Cavazos-Rehg, Melissa J Krauss, Shaina J Sowles, and Laura J Bierut. Marijuana-related posts on instagram. *Prevention Science*, 17(6):710–720, 2016.

[348] Homa Hosseinmardi, Rahat Ibn Rafiq, Richard Han, Qin Lv, and Shivakant Mishra. Prediction of cyberbullying incidents in a media-based social network. In *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 186–192. IEEE, 2016.

[349] Mokter Hossain. Crowdsourcing: Activities, incentives and users' motivations to participate. In *2012 International Conference on Innovation Management and Technology Research*, pages 501–506. IEEE, 2012.

[350] Ju Han and Kai-Kuang Ma. Fuzzy color histogram and its use in color image retrieval. *IEEE Transactions on image Processing*, 11(8):944–952, 2002.

[351] Amara Graps. An introduction to wavelets. *IEEE computational science and engineering*, 2(2):50–61, 1995.

[352] Tinghua Ai, Xiaoqiang Cheng, Pengcheng Liu, and Min Yang. A shape analysis and template matching of building features by the fourier transform method. *Computers, Environment and Urban Systems*, 41:219–233, 2013.

[353] Byung-Woo Hong and Stefano Soatto. Shape matching using multiscale integral invariants. *IEEE transactions on pattern analysis and machine intelligence*, 37(1):151–160, 2014.

[354] Itzhak Fogel and Dov Sagi. Gabor filters as texture discriminator. *Biological cybernetics*, 61(2):103–113, 1989.

[355] Li Liu, Songyang Lao, Paul W Fieguth, Yulan Guo, Xiaogang Wang, and Matti Pietikäinen. Median robust extended local binary pattern for texture classification. *IEEE Transactions on Image Processing*, 25(3):1368–1381, 2016.

[356] Lei Wu, Steven CH Hoi, and Nenghai Yu. Semantics-preserving bag-of-words models and applications. *IEEE Transactions on Image Processing*, 19(7):1908–1920, 2010.

[357] Teng Li, Tao Mei, In-So Kweon, and Xian-Sheng Hua. Contextual bag-of-words for visual categorization. *IEEE Transactions on Circuits and Systems for Video Technology*, 21(4):381–392, 2010.

[358] MD Hossain, Ferdous Sohel, Mohd Fairuz Shiratuddin, and Hamid Laga. A comprehensive survey of deep learning for image captioning. *ACM Computing Surveys (CSUR)*, 51(6):118, 2019.

[359] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[360] Simon Haykin. *Neural networks: a comprehensive foundation*. Prentice Hall PTR, 1994.

[361] Marti A. Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. Support vector machines. *IEEE Intelligent Systems and their applications*, 13(4):18–28, 1998.

[362] Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

[363] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.

[364] Dong Yu and Li Deng. *AUTOMATIC SPEECH RECOGNITION*. Springer, 2016.

[365] Moataz El Ayadi, Mohamed S Kamel, and Fakhri Karray. Survey on speech emotion recognition: Features, classification schemes, and databases. *Pattern Recognition*, 44(3):572–587, 2011.

[366] Kurt Jacobson, Mark B Sandler, and Benjamin Fields. Using audio analysis and network structure to identify communities in on-line social networks of artists. In *ISMIR*, pages 269–274, 2008.

[367] Danny Wyatt, Jeff Bilmes, Tanzeem Choudhury, and James A Kitts. Towards the automated social analysis of situated speech data. In *Proceedings of the 10th international conference on Ubiquitous computing*, pages 168–171. ACM, 2008.

[368] Danny Wyatt, Tanzeem Choudhury, Jeff Bilmes, and James A Kitts. Inferring colocation and conversation networks from privacy-sensitive audio with implications for computational social science. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(1):7, 2011.

[369] Alessandro Vinciarelli. Speakers role recognition in multiparty audio recordings using social network analysis and duration distribution modeling. *IEEE Transactions on Multimedia*, 9(6):1215–1226, 2007.

[370] Neha P Garg, Sarah Favre, Hugues Salamin, Dilek Hakkani Tür, and Alessandro Vinciarelli. Role recognition for meeting participants: an approach based on lexical information and social network analysis. In *Proceedings of the 16th ACM international conference on Multimedia*, pages 693–696. ACM, 2008.

[371] Alessandro Vinciarelli. Sociometry based multiparty audio recordings summarization. In *18th International Conference on Pattern Recognition (ICPR'06)*, volume 2, pages 1154–1157. IEEE, 2006.

[372] Shangfei Wang and Qiang Ji. Video affective content analysis: a survey of state-of-the-art methods. *IEEE Transactions on Affective Computing*, 6(4):410–430, 2015.

[373] IBM Big Data and Analytics Hub. The four v's of big data, 2017.

[374] Ibrahim Abaker Targio Hashem, Ibrar Yaqoob, Nor Badrul Anuar, Salimah Mokhtar, Abdullah Gani, and Samee Ullah Khan. The rise of "big data" on cloud computing: Review and open research issues. *Information systems*, 47:98–115, 2015.

[375] Shen Yin and Okyay Kaynak. Big data for modern industry: challenges and trends [point of view]. *Proceedings of the IEEE*, 103(2):143–146, 2015.

[376] Muhammad Habib ur Rehman, Chee Sun Liew, Assad Abbas, Prem Prakash Jayaraman, Teh Ying Wah, and Samee U Khan. Big data reduction methods: a survey. *Data Science and Engineering*, 1(4):265–284, 2016.

[377] Usama Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. From data mining to knowledge discovery in databases. *AI magazine*, 17(3):37–37, 1996.

[378] Oded Maimon and Lior Rokach. Data mining and knowledge discovery handbook. 2005.

[379] Steve LaValle, Eric Lesser, Rebecca Shockley, Michael S Hopkins, and Nina Kruschwitz. Big data, analytics and the path from insights to value. *MIT sloan management review*, 52(2):21, 2011.

[380] Anthony JG Hey, Stewart Tansley, Kristin M Tolle, et al. *The fourth paradigm: data-intensive scientific discovery*, volume 1. Microsoft research Redmond, WA, 2009.

[381] Foster Provost and Tom Fawcett. Data science and its relationship to big data and data-driven decision making. *Big data*, 1(1):51–59, 2013.

[382] Chun-Wei Tsai, Chin-Feng Lai, Han-Chieh Chao, and Athanasios V Vasilakos. Big data analytics: a survey. *Journal of Big data*, 2(1):21, 2015.

[383] Philip Russom et al. Big data analytics. *TDWI best practices report, fourth quarter*, 19(4):1–34, 2011.

[384] Dilpreet Singh and Chandan K Reddy. A survey on platforms for big data analytics. *Journal of big data*, 2(1):8, 2015.

[385] Liqiang Geng and Howard J Hamilton. Interestingness measures for data mining: A survey. *ACM Computing Surveys (CSUR)*, 38(3):9, 2006.

[386] Ken McGarry. A survey of interestingness measures for knowledge discovery. *The Knowledge Engineering Review*, 20(1):39–61, 2005.

[387] Abraham Silberschatz and Alexander Tuzhilin. On subjective measures of interestingness in knowledge discovery. In *KDD*, volume 95, pages 275–281, 1995.

[388] Robert James Hilderman and Howard John Hamilton. *Knowledge discovery and interestingness measures: A survey*. Citeseer, 1999.

[389] Gregory Piatetsky-Shapiro and Christopher J Matheus. The interestingness of deviations. In *Proceedings of the AAAI-94 workshop on Knowledge Discovery in Databases*, volume 1, pages 25–36, 1994.

[390] Bing Liu, Wynne Hsu, Lai-Fun Mun, and Hing-Yan Lee. Finding interesting patterns using user expectations. *IEEE Transactions on Knowledge and Data Engineering*, 11(6):817–832, 1999.

[391] Fritz Heider. Attitudes and cognitive organization. *The Journal of psychology*, 21(1):107–112, 1946.

[392] Phillip Bonacich. Power and centrality: A family of measures. *American journal of sociology*, 92(5):1170–1182, 1987.

[393] Márcia Oliveira and João Gama. An overview of social network analysis. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(2):99–115, 2012.

[394] G Nandi and A Das. A survey on using data mining techniques for online social network analysis. *International Journal of Computer Science Issues (IJCSI)*, 10(6):162, 2013.

[395] Francesco Bonchi, Carlos Castillo, Aristides Gionis, and Alejandro Jaimes. Social network analysis and mining for business applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):22, 2011.

[396] Mohammad Zuber. A survey of data mining techniques for social network analysis. *International Journal of Research in Computer Engineering & Electronics*, 3(6):1–8, 2014.

[397] Adedoyin-Olowe Mariam, Medhat Gaber Mohamed, and Stahl Frederic. A survey of data mining techniques for social network analysis. *School of Computing Science and Digital Media, Robert Gordon University*, pages 1–25, 2013.

[398] Jon M Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632, 1999.

[399] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1-7):107–117, 1998.

[400] James Allan. *Topic detection and tracking: event-based information organization*, volume 12. Springer Science & Business Media, 2012.

[401] Bo Pang, Lillian Lee, et al. Opinion mining and sentiment analysis. *Foundations and Trends® in Information Retrieval*, 2(1–2):1–135, 2008.

[402] Christopher D Manning, Christopher D Manning, and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT press, 1999.

[403] James Samuel Coleman, Elihu Katz, and Herbert Menzel. *Medical innovation: A diffusion study*. Bobbs-Merrill Co, 1966.

[404] Thomas W Valente. Network models of the diffusion of innovations. *Computational & Mathematical Organization Theory*, 2(2):163–164, 1996.

[405] Jure Leskovec, Lada A Adamic, and Bernardo A Huberman. The dynamics of viral marketing. *ACM Transactions on the Web (TWEB)*, 1(1):5, 2007.

[406] David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *Journal of the American society for information science and technology*, 58(7):1019–1031, 2007.

[407] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.

[408] Konstantin Shvachko, Hairong Kuang, Sanjay Radia, and Robert Chansler. The hadoop distributed file system. In *Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on*, pages 1–10. IEEE, 2010.

[409] Matei Zaharia, Mosharaf Chowdhury, Michael J Franklin, Scott Shenker, and Ion Stoica. Spark: Cluster computing with working sets. *HotCloud*, 10(10-10):95, 2010.

[410] Iva Sorić, Davor Dinjar, Marko Štajcer, and Dražen Oreščanin. Efficient social network analysis in big data architectures. In *2017 40th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 1397–1400. IEEE, 2017.

[411] Jakob Smedegaard Andersen and Olaf Zukunft. Evaluating the scaling of graph-algorithms for big data using graphx. In *2016 2nd International Conference on Open and Big Data (OBD)*, pages 1–8. IEEE, 2016.

[412] Reynold S Xin, Joseph E Gonzalez, Michael J Franklin, and Ion Stoica. Graphx: A resilient distributed graph system on spark. In *First International Workshop on Graph Data Management Experiences and Systems*, page 2. ACM, 2013.

[413] Joseph E Gonzalez, Reynold S Xin, Ankur Dave, Daniel Crankshaw, Michael J Franklin, and Ion Stoica. Graphx: Graph processing in a distributed dataflow framework. In *11th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 14)*, pages 599–613, 2014.

[414] André B Bondi. Characteristics of scalability and their impact on performance. In *Proceedings of the 2nd international workshop on Software and performance*, pages 195–203. ACM, 2000.

[415] Hesham El-Rewini and Mostafa Abd-El-Barr. *Advanced computer architecture and parallel processing*, volume 42. John Wiley & Sons, 2005.

[416] Neil J Gunther. A simple capacity model of massively parallel transaction systems. In *CMG-CONFERENCE-*, pages 1035–1035. COMPSCER MEASUREMENT GROUP INC, 1993.

[417] Neil J Gunther. *What is guerrilla capacity planning?* Springer, 2007.

[418] PP Jogalekar and CM Woodside. A scalability metric for distributed computing applications in telecommunications. In *Teletraffic Science and Engineering*, volume 2, pages 101–110. Elsevier, 1997.

[419] Alfred Giessler, J Haenle, Andreas König, and E Pade. Free buffer allocation—an investigation by simulation. *Computer Networks (1976)*, 2(3):191–208, 1978.

[420] David Sanchez, Oswaldo Solarte, Victor Bucheli, and Hugo Ordonez. Evaluating the scalability of big data frameworks. *Scalable Computing: Practice and Experience*, 19(3):301–307, 2018.

[421] Ananth Y Grama, Anshul Gupta, and Vipin Kumar. Isoefficiency: Measuring the scalability of parallel algorithms and architectures. *IEEE Parallel & Distributed Technology: Systems & Applications*, 1(3):12–21, 1993.

[422] Antonio Gonzalez-Pardo, Jason J. Jung, and David Camacho. Aco-based clustering for ego network analysis. *Future Generation Computer Systems*, 66:160 – 170, 2017.

[423] Jeffrey Heer, Michael Bostock, Vadim Ogievetsky, et al. A tour through the visualization zoo. *Commun. Acm*, 53(6):59–67, 2010.

[424] Panagiotis Karampelas. Visual methods and tools for social network analysis. *Encyclopedia of Social Network Analysis and Mining*, pages 2314–2327, 2014.

[425] Martin Krzywinski, Inanc Birol, Steven JM Jones, and Marco A Marra. Hive plots—rational approach to visualizing networks. *Briefings in bioinformatics*, 13(5):627–644, 2011.

[426] William JR Longabaugh. Combing the hairball with biofabric: a new approach for visualization of large networks. *BMC bioinformatics*, 13(1):275, 2012.

[427] Martin Wattenberg. Visual exploration of multivariate graphs. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 811–819. ACM, 2006.

[428] Peter Eades. A heuristic for graph drawing. *Congressus numerantium*, 42:149–160, 1984.

[429] Thomas MJ Fruchterman and Edward M Reingold. Graph drawing by force-directed placement. *Software: Practice and experience*, 21(11):1129–1164, 1991.

[430] Tomihisa Kamada, Satoru Kawai, et al. An algorithm for drawing general undirected graphs. *Information processing letters*, 31(1):7–15, 1989.

[431] Stefan Hachul and Michael Jünger. Drawing large graphs with a potential-field-based multilevel algorithm. In *International Symposium on Graph Drawing*, pages 285–295. Springer, 2004.

[432] Emden R Gansner, Yifan Hu, and Stephen North. A maxent-stress model for graph layout. *IEEE transactions on visualization and computer graphics*, 19(6):927–940, 2012.

[433] Mathieu Jacomy, Tommaso Venturini, Sebastien Heymann, and Mathieu Bastian. Forceatlas2, a continuous graph layout algorithm for handy network visualization designed for the gephi software. *PloS one*, 9(6):e98679, 2014.

[434] Yifan Hu and Lei Shi. Visualizing large graphs. *Wiley Interdisciplinary Reviews: Computational Statistics*, 7(2):115–136, 2015.

[435] Christopher Mueller, Benjamin Martin, and Andrew Lumsdaine. A comparison of vertex ordering algorithms for large graph visualization. In *2007 6th International Asia-Pacific Symposium on Visualization*, pages 141–148. IEEE, 2007.

[436] Nathalie Henry and Jean-Daniel Fekete. Matrixexplorer: a dual-representation system to explore social networks. *IEEE transactions on visualization and computer graphics*, 12(5):677–684, 2006.

[437] Nathalie Henry and Jean-Daniel Fekete. Matlink: Enhanced matrix visualization for analyzing social networks. In *IFIP Conference on Human-Computer Interaction*, pages 288–302. Springer, 2007.

[438] Siming Chen, Lijing Lin, and Xiaoru Yuan. Social media visual analytics. In *Computer Graphics Forum*, volume 36, pages 563–587. Wiley Online Library, 2017.

[439] Georgios A Pavlopoulos, David Paez-Espino, Nikos C Kyrpides, and Ioannis Iliopoulos. Empirical comparison of visualization tools for larger-scale network analysis. *Advances in bioinformatics*, 2017, 2017.

[440] Ben Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *The craft of information visualization*, pages 364–371. Elsevier, 2003.

[441] Jacques Bertin, William J Berg, and Howard Wainer. *Semiology of graphics: diagrams, networks, maps*, volume 1. University of Wisconsin press Madison, 1983.

[442] Bart Braden. The surveyor's area formula. *The College Mathematics Journal*, 17(4):326–337, 1986.

[443] Gabor Csardi, Tamas Nepusz, et al. The igraph software package for complex network research. *InterJournal, Complex Systems*, 1695(5):1–9, 2006.

[444] Jans Aasman. Allegro graph: Rdf triple database. *Cidade: Oakland Franz Incorporated*, 17, 2006.

[445] Ignacio Alvarez-Hamelin, Luca Dall'Asta, Alain Barrat, and Alessandro Vespignani. Lanet-vi in a nutshell, 2006.

[446] Jure Leskovec and Rok Sosič. Snap: A general-purpose network analysis and graph-mining library. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 8(1):1, 2016.

[447] J Mark Pullen. The network workbench: network simulation software for academic investigation of internet concepts. *Computer Networks*, 32(3):365–378, 2000.

[448] P Mazzuca. Y. t,"circulo: A community detection evaluation framework," feb. 2015.

[449] Paul Shannon, Andrew Markiel, Owen Ozier, Nitin S Baliga, Jonathan T Wang, Daniel Ramage, Nada Amin, Benno Schwikowski, and Trey Ideker. Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome research*, 13(11):2498–2504, 2003.

[450] Joshua O'Madadhain, Danyel Fisher, Scott White, and Y Boey. The jung (java universal network/graph) framework. *University of California, Irvine, California*, 2003.

[451] Kajdanowicz T. Bartusiak R. Sparklinggraph: large scale, distributed graph processing made easy. Manuscript in preparation, 2017.

[452] Aric Hagberg, Pieter Swart, and Daniel S Chult. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.

[453] Vladimir Batagelj and Andrej Mrvar. Pajek—analysis and visualization of large networks. In *Graph drawing software*, pages 77–103. Springer, 2004.

[454] Mathieu Bastian, Sebastien Heymann, and Mathieu Jacomy. Gephi: an open source software for exploring and manipulating networks. In *Third international AAAI conference on weblogs and social media*, 2009.

[455] Stephen P Borgatti, Martin G Everett, and Linton C Freeman. Ucinet. *Encyclopedia of social network analysis and mining*, pages 2261–2267, 2014.

[456] Jeffrey Heer, Stuart K Card, and James A Landay. Prefuse: a toolkit for interactive information visualization. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 421–430. ACM, 2005.

[457] John Ellson, Emden Gansner, Lefteris Koutsofios, Stephen C North, and Gordon Woodhull. Graphviz—open source graph drawing tools. In *International Symposium on Graph Drawing*, pages 483–484. Springer, 2001.

[458] Mahesh Lal. *Neo4j graph data modeling*. Packt Publishing Ltd, 2015.

Publicación R2 (versión sin el formato de la revista)

A Multi-Objective Genetic Algorithm for Detecting Dynamic Communities using a Local Search driven Immigrant's Scheme

Angel Panizo-LLedot
angel.panizo@upm.es
Departmento de Sistemas Informáticos
Universidad Politécnica de Madrid, Spain

Gema Bello-Orgaz
gema.borgaz@upm.es
Departmento de Sistemas Informáticos
Universidad Politécnica de Madrid, Spain

David Camacho
david.camacho@upm.es
Departmento de Sistemas Informáticos
Universidad Politécnica de Madrid, Spain

Abstract

The interest in Community Detection Problems on networks that evolves over time has experienced an increasing attention over the last years. Multi-Objective Genetic Algorithms and other bio-inspired methods have been successfully applied to tackle the community finding problem in static networks. Although, there are a large number of evolutionary and bio-inspired approaches that combine Local Search Strategies and other techniques from graph theory to handle the community detection problems in static networks, few research has been done related to the application of these algorithms over temporal, or dynamic, networks. This work is focused on the design, implementation, and the empirical analysis of a new Multi-Objective Genetic Algorithm that combines an Immigrant's scheme with local search strategies for dynamic community detection. The main contribution of this new algorithm is to address the adaptation of these strategies to dynamic networks. On the one hand, the Immigrant's scheme motif is to reuse previously acquired information to reduce computational time. On the other hand, in a dynamic environment is possible that a valid solution became invalid due to some changes in the environment, for example, if some nodes or edges have been removed or added to the network. Therefore, the aim of the local search operator used in the new algorithm is to transform an invalid solution, due to a change happened on the network, into a valid one maintaining the highest possible quality. Finally, the proposed algorithm has been tested using several synthetic and real-world networks, and compared against several algorithms (DYNMOGA, ALPA, Infomap) from the state of the art.

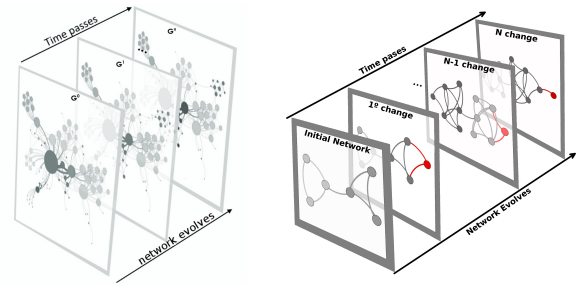
Keywords: Dynamic community finding, Multi-Objective Genetic algorithms, Graph computing, Network analysis

1 Introduction

Nowadays, the Community Detection Problem (CDP) is a highly relevant research topic due to several real-world problems can be modeled as a complex network [11]. It has been successfully applied to disciplines such as marketing [**?** 4], public healthcare [3], cybercrime [24, 25, 30] or ego-based social network analysis [15, 26], among others. There are several definitions of a community in the related literature, however, many of them are typically based on the topology information of the graph or network considered [12].

Regarding the graph connectivity, a good quality community should be connected. This means that there should always exist paths connecting each pair of vertices within the community. It is generally accepted that a subset of vertices forms a good community if the induced sub-graph is *dense*, and if there exist *connections* from the included vertices to the rest of the graph [21]. Considering both features (density and connectivity), a possible definition of a graph cluster (or community) could be a *connected component* or a *maximal clique*, i.e., a type of sub-graph in which no vertex can be added without losing the *clique property* (a clique is a subset of vertices of an undirected graph such that every two distinct vertices in the clique are adjacent, that is, its induced subgraph is complete).

In many real-world networks, such as social networks or mobile communications, the network has a temporal nature



(a) (b)

Figure 1: Subfigure a) shows a dynamic network modeled as a sequence of snapshots, each snapshot represents the whole networks in a given point of time. Subfigure b) presents a second approach, where the temporal network is modeling as an initial network and a sequence of changes.

and their structure and topology change over time. This evolution of the graph or network can be defined as a ***dynamic network*** that is usually modeled using two different approaches. As shown in Figure 1(a), one of these approaches is based on the idea of representing the network as a sequence of snapshots, where each snapshot defines the graph at in a given point of time. The second approach presents the dynamic network as an initial graph or network, and a sequence of changes of this network during the time (see Figure 1(b)). In the second approach, each element of the sequence only contains the information of the part of the network that has changed (nodes and edges that change at every instant in time), in contrast to the first model, where each element of the sequence represents the whole network at this temporal instant [1].

The communities found in these types of networks are called ***dynamic communities***, these communities are sets of nodes and edges changing over time at the same time as the structure of the network. According to the two types of *dynamic networks* models previously mentioned, there exist two different approaches that are used to represent these communities [1]. On the one hand, the first approach represents the dynamic communities as a sequence of sets of static communities (one set per snapshot). On the other hand, the second approach defines an initial static set of communities, representing the partition of the network at the initial instant of the network, to later store only the sequence of changes (nodes and edges) store the set of changes that appear on the network over time (one set of changes per temporal event).

Regarding the methods for detecting dynamic communities, there are two main issues that these methods have to address: grouping nodes for each snapshot, and tracking these groups over time. In the related literature, some works are only focused on solving one of the issues [6, 16], and other research works try to address both problems simultaneously [45, 29]. However, methods that only address the tracking of communities over time, assume that those communities have already been identified. Therefore, these types of methods have to be supported by other methods that perform the task of grouping and provide their resulting communities for each snapshot as input to them in order to be applied.

In addition, some of these methods usually take into account another important feature in these dynamic environments, which is the type of change that occurs over time. In the particular case of community detection, this change is related to the differences that occur between the different sets of communities identified. There are a group of methods which assumes that no abrupt changes occur between sets of communities in a short period of time, therefore they look for *temporal smoothness* [7] in the dynamic communities. Hence, regarding to this aspect, the dynamic detection algorithms could be classified as *non-temporal smooth methods* [45] and *temporal smooth methods* [29], depending on whether or not they address this specific issue.

In order to obtain improved algorithms for community detection problems, approaches based on evolutionary computation, such as Genetic Algorithms (GAs), have been designed to manage both static [**?** 2, 5] and dynamic [14] networks. There are several works focused on adapting GAs to dynamic environments. Their purpose is to maintain as much as possible the best solutions found after a change occurs, in order to decrease the computational effort. Some of the techniques used for this purpose are: *hypermutation* [8] (increases the mutation rate of the population when a change occurs in the environment), *memory schemes* [44] (maintains a memory of individuals during all the GA execution, and when a change occurs, use this memory to influence the actual population), and *immigrant's* [22] (inserts new individuals into the population when a change occurs in order to improve the population's diversity). However, few research work has been done on the application of previous approaches for dynamic community finding in networks [22].

In the related literature, there are several GAs approaches that has been combined with *local search strategies* for static community detection in order to improve their performance. The *local search strategies* are applied mostly during the *population initialization* [31] or during the *mutation process* [28]. In this work a new MOGA algorithm, based on NSGAII [10], is proposed. The new algorithm joins these two approaches and tries to take advantage of both. The new method proposed uses an immigrant's scheme specifically customized to the community detection problem in temporal networks. The aim of the immigrant scheme is to adapt the MOGA algorithm to dynamic environments increasing the diversity capacity of the

algorithm. Finally, the new algorithm combines this immigrant's scheme with a *local search* to improve its performance.

The main idea behind applying these two techniques is that consecutive snapshots in a dynamic network have a similar topology, so the results obtained for one snapshot can be reused by the next one. Therefore, the proposed method runs the MOGA for each snapshot on a dynamic network to identify its communities, but the initial population for each run of the MOGA is generated using a *local search* method. In particular, this *local search* method is applied over the resulting population from the previous snapshot, to later inserting the new individuals as immigrant's. In a dynamic environment is possible that a valid solution became invalid due to some changes in the environment, for example, where some nodes or edges have been removed or added to the network. Therefore, the aim of the local search operator used in the new proposed approach is not improving a valid solution, but to transform an invalid solution due to a change happened on the network, into a valid one maintaining the highest possible quality. Finally, the proposed algorithm has been tested using several synthetic and real-world networks, and its results have been compared against a subset of well-known algorithms from the state of the art in the area of CDP. The work in this article further expand the results published in [35] and [36].

The main contributions of this work can be summarized as follows:

- The design and testing of three different *local search* methods (label propagation, random walks, and a hybrid approach that combines previous local search methods) to generate individuals with good quality to be used in the immigrant's scheme.
- The validation of the algorithm by comparing it against a subset of well-known algorithms from the state of the art in this topic.
- The use of a Multi-Objective approach for detecting dynamic communities provides a Pareto Front of solutions (network partitionings) for each snapshot instead of a single solution. The detection of several solutions for each snapshot in the grouping phase will allow us to feed the tracking algorithms with more information with the aim of improving its performance.

The rest of the paper is structured as follows: Section 2 reviews other related methods for community finding. Section 3 describes the dynamic community detection problem and the proposed algorithm. Sections 4 and 5 present the procedure followed to test the algorithm and discuss the experimental results. Finally, the last section draws some conclusions and future research lines of work.

2 Background

2.1 Problem Formulation

Let $\aleph = \{G^0..G^n\}$ be a dynamic network with n snapshots sorted in chronological order. Each snapshot G^t is modeled as a graph $G^t = (V^t, E^t)$, where V^t is a set of vertices, called nodes, and E^t is a set of links, called edges, that connects two elements of V^t at time t. Then, the goal is to group the vertices for each G^t in such a way that vertices in the same group have more edges among them than with the rest of vertices of G^t. In this work we will focus on non-overlapping communities, so each vertex or node in G^t can only belong to one single group.

2.2 Methodologies for detecting dynamic communities

There are two main issues in dynamic communities detection problems: grouping nodes from each snapshot, and tracking these groups over time. Due to these issues, four main approaches have been used in the literature to detect dynamic communities [1]:

1. *Independent static community detection and matching*: The first methods that emerged to address this problem are based on an approach consisting of two independent steps. In this approach, firstly the static communities for each snapshot are identified, and then the matching between communities belonging to consecutive snapshots are performed [20, 34]. To carry out the first step of this methodology, classic methods for static community finding can be directly applied. However, the main drawback of this approach is the unstable nature of the communities detected between different snapshots. Most of the more efficient static community detection algorithms are stochastic, and two runs on the same network as input do not necessarily provide the same solution. Therefore, these methods are more prone to generate noisy results.

2. *Community detection by informed iterative static community detection*: In order to address the problem of instability in the identified communities, a new approach arose, where only the static communities are detected on the first snapshot. The rest of snapshots are processed following a chronological order, and using the communities detected at the previous snapshot to identify the communities at current snapshot [29, 7]. These type of approaches solves the problem of the instability of communities, but they have a dependence

on data between snapshots. They still need to integrate a matching phase that allows tracking communities between snapshots. Most of the proposed methods in the literature avoid this phase by forcing a fixed number of nodes.

3. *Global Community detection on all snapshots*: These methods process all the sequence of snapshots simultaneously, applying a single community detection algorithm to generate a set of communities which contains nodes from different snapshots [32, 48].One of the main problems of this approach is its very high computational cost when are applied to large scale temporary networks. Moreover, it cannot be applied to networks evolving in real time.

4. *Community detection using temporal networks*: This approach is based on the idea of finding the static communities in the first snapshot. Then for the rest of snapshots, only the modifications of the communities between them are deduced [47, 17, 45]. Analyzing the main characteristics of this type of approaches, it can be seen that its main advantage is the decrease in complexity because only changes in the network between snapshots are addressed, rather than entire networks. However, the drawback of this solution is the difficulty to ensure a long-term coherence of the dynamic communities, due to a complete community detection on a whole static network is never run after the first step.

2.3 Evolutionary Approaches

In the related literature, there are also several *bio-inspired approaches* to address the issue of the detection of dynamic communities. For example, Folino and Pizzuti present the DYMONGA [14] algorithm which is a method focused on the second approach shown above, and based on a multi-objective GA (MOGA). DYMONGA only addresses the grouping of nodes in each snapshot, without taking into account the evolution of those communities.

The work presented by Kim et al. [22] also proposes a MOGA approach, but in this case they follow the third approach presented, to achieve the best tracking of the partition of the network over time. In contrast to DYMONGA, instead of returning partitions for each snapshot, this method returns the partition that best fits the last snapshot, taking into account all previous snapshots.

In order to improve the performance of these GA approaches for detecting communities, there are several works combining these methods with a local search. For example, Li et al.[28] introduce a new mutation operator that uses a local search based on the small-world phenomena to apply

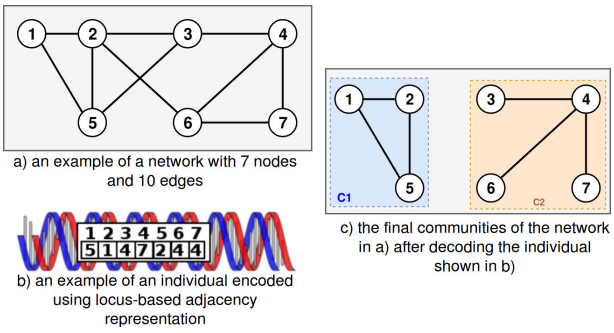


Figure 2: Locus-based adjacency encoding. This figure shows an example of an individual encoding using a locus-based adjacency and its resulting set of communities.

the operator in the evolutive process. In this regard, Wang et al. [46] proposed a modified crossover operator more suitable for community detection, and a heuristic mutation operator based on local modularity. Finally, Pizzuti [39] introduced another approach in which a local search is applied once the GA process is ended, to improve the quality of the community. All these hybrid approaches combining a GA with a Local Search, handle only static networks (networks that do not change over time). Their main objective is taking an already valid solution and improving it as much as possible. Therefore, these approaches should be previously adapted to be later applied on dynamic networks.

3 Algorithm Design

This section introduces the proposed algorithm for dynamic community detection. The new algorithm receives as input a dynamic network modeled as a sequence of snapshots. The final result will be a set of communities for each one of these snapshots. The structure and dynamics of the algorithm are shown in Figure 3.

The algorithm uses a "locus-based adjacency" encoding [37], where each individual represents one possible solution (set of communities) for a particular snapshot. In this encoding, a particular allele corresponds to a node of the graph. Therefore, given a graph $G = (V, E)$, an individual of the population will have a size equal to m, where m represents the total number of nodes in the graph ($m = |V|$). Figure 2 (a) illustrates a graph, and a possible example of a chromosome according to this representation (b). The position of each allele (i) corresponds to a particular node, and its value can be inferred from the adjacency matrix of this node (this value is randomly selected from the set of the adjacent nodes). For example, in Figure 2 (b), the adjacent nodes to the node 1 are $\{2, 5\}$, and in this case the value assigned is 5. A value of j assigned to the i position is interpreted as a link between the nodes i and j, and in the

resulting solution, they will be in the same cluster.

Finally, a decoding process is needed to identify all the communities. In this process, the nodes belonging to the same connected component are then assigned to the same community. Continuing with the example shown in Figure 2, if the process starts on the first allele which implies that node 1 (position of the allele) and node 5 (value of the allele) belong to the same community. The next position of the allele shows that node 2 and 1 belongs to the same cluster. However, node 1 is already in a previous community, so node 2 is included into the existing previous community. Following this process for all the positions the resulting communities are those shown in Figure 2 (c).

Regarding the algorithm's structure (see Figure 3), it starts generating an initial random population using the first snapshot (G^0) of the dynamic network (module "make random population"). To create this random initial population, each gene for each individual that corresponds to a node of the graph is initialized by selecting a value from the set of the adjacent nodes to it. Then, the population undergoes a multi-objective evolutionary process (module called "NSGA-II") based on the standard NSGA-II algorithm [10]. Finally, once the evolution process has finished, the resulting Pareto Front is analyzed in order to select one individual to be used as a final solution. For the successive snapshots of the network ($G^t|t > 0$), instead of generating a new initial random population, the last population from the previous step will be used, inserting individuals of it as immigrants to the new one. Due to changes in the network, is possible that the individuals in the last population are no longer valid individuals in the actual snapshot. Hence, it is necessary to check and **repair**, when needed, all the immigrants inserted into the population before running the evolutionary process again (module "repair population"). Finally, these three steps of the algorithm (repair population, NSGA-II, and select solution) are repeated until all snapshots have been processed.

The next subsections describe in detail the three main parts of the proposed algorithm: I) the configuration used for the NSGA-II algorithm, II) the method for preserving information between consecutive snapshots, and III) the method for choosing the individual of the Pareto Front to be used as a solution.

3.1 NSGA-II configuration

To begin with, the fitness function implemented in the algorithm minimizes two different objective functions: the "Inverse Community Score"[38] and the "Average ODF" [13]. These two objective functions have been selected following the guidelines published in [43].

The Inverse Community Score measures how well interconnected the nodes of a community are. This score is

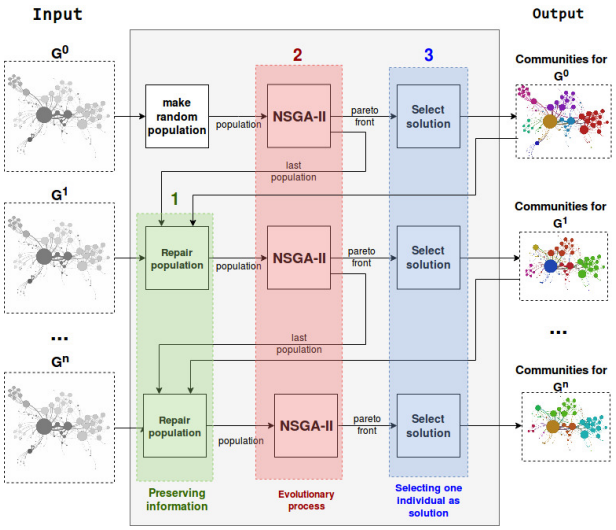


Figure 3: Diagram of the algorithm proposed for dynamic community detection. It receives a dynamic network modeled as a sequence of snapshots as input, and returns the communities for each snapshot as the final solution.

defined as follows:

$$C_SCORE^{-1} = \frac{1}{\sum_{S \in C}(\frac{2ms}{ns})^2} \quad (1)$$

where C is the set of communities that an individual encodes; S is each of those communities; ms is the number of edges inside a community S and ns is the number of nodes in community S.

Its maximum value is ∞ and is given when each node forms a community with only one member. Its minimum value is $1/(n-1)^2$, where n is the number of nodes in the graph. The minimum value is obtained when all the nodes in the network form a clique, so they will be members of the same (unique) community.

Regarding the Average ODF metric, it measures the fraction of edges that point outside a community as follows:

$$AVG_ODF = \sum_{S \in C}(\frac{1}{ns}\sum_{u \in S}(\frac{|\{u, v|v \notin S\}|}{d(u)})) \quad (2)$$

where C is the set of communities that an individual encodes; S is each of those communities; ns is the number of nodes in the community S and $d(u)$ is the degree of node u.

Its minimum value is 0 and is given when there are no connections between communities. Its maximum value is n, where n is the number of nodes in the graph, and is obtained when each node forms a community composed only by itself.

Related to the operators used in NSGA-II, we are using a custom function that randomly selects several genes of an individual and changes their values to a random neighbor of the node associated with that particular gene as mutation operator. For generating new offspring, we are using a two-point crossover operation. As the selection operator, we are using a tournament of size 2. Finally, as stop criteria of the evolution process, the hypervolume [49] of the Pareto Front is used. If the hypervolume stays unchanged for a number N of generations (this value will be experimentally fixed), or a maximum number of generations is reached, the algorithm stops.

3.2 Preserving past information

The mechanism for preserving information between consecutive snapshots of the network (see block 1, repair information module, in Figure 3) is based on the idea that consecutive snapshots in a dynamic network have a similar topology. So the results obtained for one snapshot (last population) can be reused by the next one (initial population). However in a dynamic environment is possible that a valid solution became invalid due to some changes in the environment, for example, that some nodes or edges have been removed or added to the network. Therefore, this mechanism is in charge of updating the immigrant individuals (repair population) from previous population to adapt them to the changes that a network suffers between two consecutive snapshots. The main goal of this process is generate an initial population of better quality individuals that allows speeding up the evolutionary process.

Algorithm 1 Repair population algorithm

```
1: function REPAIR_POPULATION(P_best^{t-1}, G^t, C^{t-1})
2:     for all Ind ∈ P_best do
3:         updateIndLength(G^t.Nodes, Ind)
4:         for all Gen_i ∈ [0, length(Ind)] do
5:             Gen_v ← Ind[Gen_i]
6:             if !Gen_v ∨ !connected(G^t, Gen_i, Gen_v) then
7:                 n ← localSearch(Gen_i, G^t, C^{t-1})
8:                 Ind[Gen_i] ← n
9:     return P_best
```

The Algorithm 1 shows the pseudo-code of the repair process applied. This method receives as input the population generated by the last evolutionary process (P_{best}^{t-1}), the actual snapshot (G^t) and the set of communities selected as a solution for the previous snapshot (C^{t-1}). The repair process starts by updating the length of the individuals of the previous population (line 3 of the algorithm) because if nodes have appeared or disappeared this size would be different. This method receives an individual (Ind) and the nodes of the actual snapshot (G^t). The method removes from the individual the genes that were related to nodes that have disappeared from the snapshot. And it also adds new genes to the individual for representing the new nodes that have appeared in the current snapshot. The value of these new genes added to an individual is undefined, so the repair process checks the values of all its genes (lines 4-8). If a gene is undefined or its value corresponds to a node that is no longer a neighbor of the node related to that gene (line 6), a local search method is applied to decide the gene's new value (lines 7-8). In this work, we propose the use of three local search methods: i) based in Label Propagation [40], ii) based in Random Walks [41], iii) and a combination of the two previous strategies.

3.2.1 Local Search based on Label Propagation

The first local search method proposed in this work is based on Label Propagation. This method repairs an invalid or undefined gene (node) by assigning it to the most repeated community among its neighbors. As shown in Algorithm 2, the method receives as input the node to repair ($Node_i$), the current snapshot (G^t), and the communities found for the previous snapshot (C^{t-1}). It starts obtaining all the neighbors of the node to repair (line 2). Then, the most repeated community between the neighbors (line 3) is selected. Next, the neighbors not belonging to the most repeated community (line 4) are filtered. Finally, one node is selected randomly from the remaining neighbors (line 5) to be assigned as the new value of the gene.

Algorithm 2 Local search method based on Label Propagation

```
1: function LABEL_PROPAGATION(Node_i, G^t, C^{t-1})
2:     neighs ← getNeighbors(Node_i, G^t)
3:     comm ← communityMost(neighs, C^{t-1})
4:     validNodes ← nodesInComm(neighs, comm)
5:     return selectRandom(validNodes)
```

3.2.2 Local Search based on Random Walks

The second local search method proposed is based on Random Walks [41]. This method launches several random walkers in order to decide the community of an invalid or undefined gene (node). After all the random walkers have finished, the most frequently visited community is selected to be assigned to the gene (node) to be repaired. A random walker moves randomly between the nodes on a graph for a fixed number of steps. The walker starts on a node and in each step moves to one of its neighbors. All the neighbors have the same probability of being selected. The walker continues until a fixed number of steps has been reached.

Algorithm 3 shows that this method receives as input the node being repaired ($Node_i$), the current snapshot (G^t), and the communities found for the previous snapshot (C^{t-1}). In addition, the method has two configurable parameters: $num_walkers$ and num_steps. The first parameter sets the number of random walkers dispatched. The latter, the number of steps that each walker will perform. The method begins by launching $num_walkers$ random walkers (lines 3-5). Each walker will start on the node $Node_i$ and will perform its random walks (line 4) until the walker has performed num_steps steps. The information about all the nodes visited, and how many times the walker has visited them (line 5), is stored. Once all the walkers have finished, this information is used to check the most frequently visited community (line 6). Then, all the neighbors of the node belonging to the selected community (lines 7-8) are selected, and one of them is randomly chosen (line 9) to be returned as the new value of the gene. To save computation time the most frequently visited community is stored on a cache. Thus, the random walkers only have to be launched once per repaired node.

Algorithm 3 Local search method based on Random Walks

```
1: function RANDOM_WALKS(Node_i, G^t, C^{t-1})
2:     walks ← null
3:     for all i ∈ [0, num_walkers] do
4:         walk_i ← doWalk(Node_i, G^t, num_steps)
5:         update(walks, walk_i)
6:     comm ← communityStayMost(walks, C^{t-1})
7:     neighs ← getNeighbors(Node_i, G^t)
8:     validNodes ← nodesInComm(neighs, comm)
9:     return selectRandom(validNodes)
```

3.2.3 Hybrid Local Search

The third and last method is a combination of the other two methods previously presented. Algorithm 4 shows the pseudo-code for this method. As with Algorithm 1, the method receives as input the population generated by the last evolutionary step (P_{best}^{t-1}), the actual snapshot (G^t), and the community set selected as a solution for the previous snapshot (C^{t-1}). It starts creating two local populations named P_{label} and P_{walks} (lines 2-3). The first local copy stores the individuals repaired by the Label Propagation method. The latter, stores the individuals repaired by the Random Walks one. For each individual in P_{best}^{t-1}, the method first updates its length (line 5) and then generates two local copies named I_{label} and I_{walks} (lines 6-7). On the one hand, the genes of the copy I_{label} are subdued to the Label Propagation repair method (lines 11-12). On the other hand, the genes of the copy I_{walks} are subdued to the Random Walks one (lines 13-14). Once all the genes in an individual have been processed, both local copies are stored in their respective populations (lines 15-16). Finally, after all the individuals in the population have been processed, the non-dominated sorting selection method of NSGA-II (see block 2, NSGA-II module, in Figure 3) is used over the combination of both local populations to choose the individuals returned (line 17).

Algorithm 4 Hybrid Local search method

```
1: function HYBRID_REPARATIONS(P_best^{t-1}, G^t, C^{t-1})
2:     P_label ← ∅
3:     P_walks ← ∅
4:     for all Ind ∈ P_best do
5:         updateIndLength(G^t.Nodes, Ind)
6:         I_label ← copy(Ind)
7:         I_walks ← copy(Ind)
8:         for all Gen_i ∈ [0, length(Ind)] do
9:             Gen_v ← Ind[Gen_i]
10:            if !Gen_v ∨ !connected(G^t, Gen_i, Gen_v) then
11:                n ← label_propagation(Gen_i, G^t, C^{t-1})
12:                I_label[Gen_i] ← n
13:                n ← random_walks(Gen_i, G^t, C^{t-1})
14:                I_walks[Gen_i] ← n
15:        P_label ← P_label ∪ I_label
16:        P_walks ← P_walks ∪ I_walks
17:    return selNSGAII(P_label ∪ P_walks, length(P_best))
```

3.3 Selecting one individual as solution

The last step of the proposed method is selecting one individual from the Pareto front as the resulting solution of the snapshot (see block 3, Select solution module, in Figure 3). For this purpose, the Modularity [33] for each individual in the Pareto front is calculated to measure the quality of the communities that represent each of them. The individual with a higher Modularity is chosen as the solution for that snapshot. Algorithm 5 shows the process followed for this method. Its receives a Pareto front (P_{pareto}), and the current snapshot (G^t). Finally, the individual selected as the solution for that snapshot is returned.

4 Algorithm Tuning

Section 3 presents several methods to preserve information after a change has occurred. In order to evaluate the performance of these methods, a comparative assessment of the Hypervolume is carried out using several synthetic

Algorithm 5 Select solution

```
1: function SELECT_SOLUTION(P_pareto, G^t)
2:     I_solution ← NULL
3:     M_best ← −∞
4:     for all I ∈ P_pareto do
5:         M_local ← modularity(I, G^t)
6:         if M_local ≥ M_best then
7:             I_solution ← I
8:             M_best ← M_local
9:     return I_solution
```

dynamic networks to generate an appropriate data benchmark. The following subsections describe the tool used to generate the synthetic networks, the specific datasets used for the experiments and finally, the results obtained.

The algorithms proposed in this work have been developed in Python 3.6 using the implementation of NSGA-II available on DEAP[1] library. Also, all the experimentation in this work were done on an Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20GHz computer with 125 GB of RAM.

4.1 Evaluation Benchmark

For generating the synthetic dynamic networks, we have used the DANCER tool [27]. DANCER generates synthetic dynamic attributed networks with community structure that follows the properties of real-world networks. This tool models the dynamics of the network using two kinds of operations: Micro-operations and Macro-operations. On the one hand, Micro-operations works with nodes and edges, adding or removing them. On the other hand, Macro-operations works with whole communities, merging or splitting them.

To test the proposed techniques, 66 datasets have been generated using the DANCER workbench. Each dataset uses a different combination of Micro and Macro operations in order to test networks with different characteristics. Obviously, dynamic networks that have fewer changes between snapshots are easier to process. Therefore, we have used the graph edit distance [42] as a measure of the complexity of a dynamic network. The graph edit distance compares two consecutive snapshots in a network and counts the number of nodes/edges additions and deletions needed in order to transform one of the graphs into the other. Its minimum value is 0 and this happens when both graphs are identical. On the other hand, its maximum value is $n_a + e_a + n_b + e_b$, where n_a and n_b are the number of nodes of the first and second graphs respectively and e_a and e_b are the number of edges. All the dynamic networks generated are modeled as a sequence of snapshots and are composed of 10 snapshots.

The 66 datasets have been split into several categories in a hierarchical way. First, we have separated the datasets that always contains the same nodes and only the edges change from the datasets that change the nodes and edges. In addition, each category has been further divided into datasets that perform only micro-operations and datasets that perform both, micro and macro operations. Furthermore, the datasets in both categories have been generated in a progressive way, starting with datasets that only perform one operation to datasets that performs all the operations combined. Also, for each operation, we have considered several variations, for example, in macro-operations, we have made datasets where one operation occurs more often than another, or as another example, in micro-operations, we have considered that an edge is inserted inter-communities or intra-community. Finally, for each category, three datasets with increasing dynamism have been generated. In order to achieve increasing dynamism, the number of operations performed is increased. These datasets contain the prefixes "01", "02" and "03". This hierarchical division allows us to evaluate each operation isolated and discern if some of the local search operators fail to properly work in some particular cases.

Tables 1 and 2 shows all the datasets generated for testing the different techniques. On the one hand, Table 1 shows the datasets that maintain the same nodes over all the snapshots and only the edges change. On the other hand, Table 2 shows those datasets that contains snapshots where nodes have been added or removed. Each table is separated into 3 columns. The first column contains the name of the dataset and a brief description of the operations performed. The second column shows several variations, when available, of the same operations. For example, when adding edges, an edge can be added inside a community, which makes the community more obvious and the resulting network easier to process. Contrary, if the edge is added connecting two different communities, both communities became more blurry and the network becomes harder to process. The third column is likewise divided into three. Each of the sub-columns indicates the number of operations performed. Column 01 contains the benchmarks with the lowest number of operations performed, whereas column 03 contains the benchmark with the highest number of operations performed, column 02 contains the benchmarks with where the number of operations falls between the ones in column 01 and column 02. Note the increasing graph edit distance in these three columns. Lastly, for reproducibility issues all these data sets have been openly published on a Kaggle repository[2].

[1] https://deap.readthedocs.io/en/master/

[2] https://www.kaggle.com/apanizolledot/dancer-synthetic-dynamic-networks

Table 1: Table of dynamic networks with fixed number of nodes for testing.

Dataset	**Variation**	**Graph Edit** 01	02	03
Micro operations				
fn_ge: Only gains edges from one snapshot to the next	**c+**: gains edges within a community more often	36	75	128
	d+: gains edges joining two communities together more often	30	64	75
	eq: both types of edges have same probability of being added	111	121	233
fn_le: Only losses edges from one snapshot to the next	**c+**: losses edges joining two communities together more often	34	64	87
	d+: losses edges within a community more often	47	54	89
	eq: all edges have same propability of being removed	78	110	149
fn_mx: Gains and losses edges from one snapshot to the next	**c+**: More edges are added inside communities than between them	87	265	476
	d+: More edges are added between communities than inside	71	218	437
	eq: no distinction is made	169	319	390
Micro and Macro operations				
fn_sp: Communities split	NA	27	49	03
fn_mg: Communities merge	NA	31	31	51
fn_sm: Communities split and merge	**eq**: Merge and split occurs with the same probability	62	75	100
	m+: Merge occurs more often	25	52	86
	s+: Split occurs more often	52	39	118

Table 2: Table of dynamic networks with variable number of nodes for testing.

Dataset	**Descrition**	**Graph Edit** 01	02	03
Micro operations				
ln	Only losses nodes	80	90	181
gn	Only gains nodes	313	743	1249
mx	Gains and losses nodes	429	1014	1328
Micro and Macro operations				
sp	Communities splits in two	990	1120	1267
mg	Communities merge together	929	1149	1203
Dataset	**Variation**	**Graph Edit** 01	02	03
sm: Communities split in two and merge together	**eq**: Merge and split occurs with the same probability	1159	1203	1255
	m+: Merge occurs more often than Split	1054	1300	1157
	s+: Split occurs more often than Merge	943	1195	1390

4.2 Reparation Technique Selection

In this section, the results of the tests performed to select the most suitable reparation technique are discussed. In order to assess the improvement provided by the use of a reparation technique, first, the proposed method is run without applying the immigrant's strategy. Hereafter we will call this algorithm as *Standard*. For all the tests in this article, the Standard algorithm is used as a mean of comparison. After that, each of the populations generated by the Standard method, one per snapshot, are repaired using the proposed reparation techniques. Finally, we measure the hypervolume and Modularity of the populations resulting from applying the reparation technique to the populations generated by the Standard method. The approach that generates populations with higher hypervolume indicates that their individuals (solutions) are better distributed among the Pareto Front, hence the population has a higher quality. When two methods output similar hypervolume, the one with better Modularity will be selected. Notice that we have favor hypervolume over Modularity, in our opinion as the Pareto Fronts generated in this stage still have not been subjected to the evolutionary process is more desirable to have a more diverse population over a population with a few better individuals. Due all of our methods are stochastic, each experiment has been repeated 20 times. In addition, a Kruskal-Wallis H-test [23] has been carried out, among the results obtained by the three techniques and for each dataset, to check that the results are statistically significant.

Tables 3 and 4 shows the sum of the Hypervolumes obtained at each snapshot for each dataset and each reparation technique applied. This first Table contains the results for datasets that have a fixed number of nodes, the latter, for datasets that have a variable number of nodes. Each value of the Hypervolume shows the mean and the standard deviation for the 20 executions of the proposed method for each reparation technique of a dataset. A value is marked with a tick(✓) when the Kruskal-Wallis H-test returns a p_value smaller than 0.05. Otherwise, a cell is marked with a cross(✗). When the results are different, the one that gives better performance is shown in bold numbers. Finally, the two user-defined parameters ($num_walkers$ and num_steps) used in the Random Walks operator have been set to 6 and 5 respectively. These values have been experimentally tuned.

Analyzing Table 3, we can see that in most cases both reparation techniques show a similar performance. Nevertheless, there are some datasets where the Label Propagation technique obtains better results than the Random Walks one and vice-versa. Regarding the performance of the hybrid technique, we can see that in some few datasets, like fn_mx_c+_02, this technique shows better results than any of the other two techniques alone. However, it can be noticed that the performance difference between the three strategies is low. On the other hand, analyzing the results shown in Table 4, it can be seen that for networks where the nodes change between snapshots, the Random Walks reparation technique is better than the Label Propagation one. Moreover, using the combination of both methods does not give a better performance than using only the Random Walks one. In addition, it can be noticed, that in this case, the difference in performance between methods is more remarkable.

In order to deeply analyze this phenomenon, we have studied the performance of each operator at the snapshot level instead of at the whole dynamic network level. These studies have shown that the difference in performance between the Label Propagation method and the Random Walks ones, depends on the number of new nodes a dataset receives. To analyze this, we have selected the dataset mx_03 because during its evolution it has snapshots where only nodes are removed, snapshots where a lot of new nodes are added and very few have been deleted, and snapshots where both cases happens.

Figure 4 shows the fluctuation in the number of nodes between snapshots in dataset mx_03, and the Pareto Fronts obtained from snapshots 3, 4 and 2 in the same dataset. In subfigure (a), the red part of the columns shows the number of nodes that have disappeared from the network, the green part shows the number of new nodes and the blue part shows the number of nodes that continue to stay in the network from the previous snapshot. In subfigures (b), (c) and (d) the red dots represent the solutions returned by the Random Walks method and the green dots the results returned by the Label Propagation one. Subfigure (b), contains the Pareto Fronts obtained from snapshot 3 of the dataset. As we can see the Pareto Fronts obtained by both methods are more or less the same, also if we look at snapshot 3 in subfigure (a) we can see that no new nodes are added. Likewise, subfigure (c) contains the Pareto Fronts obtained from snapshot 4 of the dataset. Contrary to the Paretos obtained from snapshot 3, this time the Random Walks method works better than the Label Propagation one. When looking at snapshot 4 in subfigure (a), we can see that this snapshot contains only some few new nodes (green bar). Finally, subfigure (d) contains the Paretos obtained from snapshot 2. As we can observe, in this case, the aforementioned difference in performance has been intensified. If we look at snapshot 2 in subfigure (a) we can see that the snapshot contains a higher number of new nodes compared with snapshot 4. The results obtained for this dataset are similar for the rest of them. Taking this into account, we can conclude that the more new nodes a snapshot have, the bigger the difference in performance between the Label Propagation method and the Random Walks one is.

Regarding the Modularity obtained, Tables 5 and 6 show the sum of the values obtained at each snapshot for each

Table 3: Hypervolume performance of reparation operators for datasets with a fixed number of nodes. The tick indicates that there is statistical evidence between the results to be considered different, the cross otherwise. Each cell shows the mean value and the standard deviation from 20 repetitions.

Dataset	Label Propagation	Random Walks	Hybrid	
fn_le_c+_01	8928.19 +/- 14.45	8929.44 +/- 9.74	8931.44 +/- 10.76	✗
fn_le_c+_02	8898.06 +/- 12.95	8904.38 +/- 10.88	**8914.48 +/- 8.84**	✓
fn_le_c+_03	8884.17 +/- 16.41	**8926.34 +/- 8.06**	8927.21 +/- 8.71	✓
fn_le_d+_01	8853.12 +/- 20.57	8835.49 +/- 22.15	8862.37 +/- 21.28	✗
fn_le_d+_02	8852.50 +/- 18.05	8838.32 +/- 16.51	8858.04 +/- 18.91	✗
fn_le_d+_03	**8835.88 +/- 17.09**	8819.85 +/- 11.90	8843.55 +/- 13.47	✓
fn_le_eq_01	**8904.13 +/- 14.30**	8893.76 +/- 7.51	8911.72 +/- 8.95	✓
fn_le_eq_02	**8905.63 +/- 12.30**	8877.42 +/- 15.83	8910.52 +/- 8.14	✓
fn_le_eq_03	8790.22 +/- 25.22	**8827.40 +/- 14.59**	8837.00 +/- 12.16	✓
fn_ge_c+_01	8931.46 +/- 7.68	8931.46 +/- 7.68	8931.46 +/- 7.68	✗
fn_ge_c+_02	8887.66 +/- 16.57	8887.66 +/- 16.57	8887.66 +/- 16.57	✗
fn_ge_c+_03	8906.78 +/- 14.10	8906.78 +/- 14.10	8906.78 +/- 14.10	✗
fn_ge_d+_01	8909.25 +/- 7.08	8909.25 +/- 7.08	8909.25 +/- 7.08	✗
fn_ge_d+_02	8868.06 +/- 9.28	8868.06 +/- 9.28	8868.06 +/- 9.28	✗
fn_ge_d+_03	8882.65 +/- 20.80	8882.65 +/- 20.80	8882.65 +/- 20.80	✗
fn_ge_eq_01	8905.63 +/- 6.67	8905.63 +/- 6.67	8905.63 +/- 6.67	✗
fn_ge_eq_02	8889.74 +/- 17.40	8889.74 +/- 17.40	8889.74 +/- 17.40	✗
fn_ge_eq_03	8887.93 +/- 20.80	8887.93 +/- 20.80	8887.93 +/- 20.80	✗
fn_mx_c+_01	8912.94 +/- 10.73	**8928.67 +/- 6.78**	8928.73 +/- 6.84	✓
fn_mx_c+_02	8772.25 +/- 30.59	8783.04 +/- 18.24	**8811.07 +/- 18.21**	✓
fn_mx_c+_03	8672.74 +/- 22.17	**8721.93 +/- 32.09**	8742.70 +/- 24.48	✓
fn_mx_d+_01	8849.42 +/- 19.17	8842.91 +/- 18.36	8856.44 +/- 20.78	✗
fn_mx_d+_02	8768.42 +/- 27.35	**8788.95 +/- 21.31**	8801.78 +/- 21.85	✓
fn_mx_d+_03	8642.81 +/- 24.71	8664.92 +/- 19.80	**8688.25 +/- 14.66**	✓
fn_mx_eq_01	8837.75 +/- 21.73	8830.98 +/- 18.80	8846.95 +/- 18.23	✗
fn_mx_eq_02	8662.69 +/- 24.76	8690.72 +/- 28.22	**8716.87 +/- 21.36**	✓
fn_mx_eq_03	8732.17 +/- 20.34	8737.04 +/- 12.63	**8769.48 +/- 18.45**	✓
fn_sp_01	8904.12 +/- 6.37	8901.15 +/- 6.51	8904.21 +/- 6.22	✗
fn_sp_02	8941.17 +/- 5.88	8939.07 +/- 6.59	8941.20 +/- 5.83	✗
fn_sp_03	**8923.77 +/- 7.83**	8909.12 +/- 8.75	8925.45 +/- 7.48	✓
fn_mg_01	17697.48 +/- 23.52	17697.48 +/- 23.52	17697.48 +/- 23.52	✗
fn_mg_02	17653.14 +/- 15.27	17653.14 +/- 15.27	17653.14 +/- 15.27	✗
fn_mg_03	17697.72 +/- 19.42	17697.72 +/- 19.42	17697.72 +/- 19.42	✗
fn_sm_eq_01	17730.95 +/- 20.87	17729.22 +/- 19.18	17733.68 +/- 19.59	✗
fn_sm_eq_02	17632.09 +/- 25.52	17631.75 +/- 26.01	17634.38 +/- 24.97	✗
fn_sm_eq_03	17728.30 +/- 21.54	17725.12 +/- 23.92	17729.48 +/- 22.32	✗
fn_sm_m+_01	17690.75 +/- 18.90	17692.89 +/- 20.09	17693.87 +/- 19.14	✗
fn_sm_m+_02	17638.63 +/- 16.46	17636.95 +/- 15.42	17639.04 +/- 17.23	✗
fn_sm_m+_03	17606.98 +/- 32.04	17606.67 +/- 33.27	17613.11 +/- 31.06	✗
fn_sm_s+_01	8928.07 +/- 10.81	8930.58 +/- 9.95	8932.13 +/- 8.93	✗
fn_sm_s+_02	8928.67 +/- 7.50	8926.88 +/- 9.54	8929.35 +/- 7.41	✗
fn_sm_s+_03	8868.24 +/- 13.86	8863.91 +/- 16.72	8875.24 +/- 11.89	✗

Table 4: Hypervolume performance or reparation operators for datasets with a variable number of nodes. The tick indicates that there is statistical evidence between the results to be considered different, the cross otherwise. Each cell shows the mean value and the standard deviation from 20 repetitions.

Dataset	Label Propagation	Random Walks	Hybrid	
ln_01	15481.46 +/- 17.16	15483.55 +/- 14.51	15488.66 +/- 16.06	✗
ln_02	14419.41 +/- 22.94	14416.96 +/- 20.96	14421.20 +/- 20.41	✗
ln_03	13833.64 +/- 19.30	13830.66 +/- 22.36	13834.42 +/- 19.56	✗
gn_01	12035.59 +/- 29.88	**12514.34 +/- 17.41**	12513.83 +/- 14.39	✓
gn_02	14582.52 +/- 38.76	**15663.27 +/- 20.01**	15663.43 +/- 20.81	✓
gn_03	19599.25 +/- 58.52	**21299.40 +/- 30.44**	21299.33 +/- 31.43	✓
mx_01	17462.13 +/- 24.67	**18024.79 +/- 18.07**	18027.36 +/- 20.02	✓
mx_02	20065.22 +/- 69.32	**21281.98 +/- 32.00**	21285.46 +/- 29.58	✓
mx_03	22936.06 +/- 58.21	**25013.87 +/- 42.47**	25013.57 +/- 43.35	✓
sp_01	12214.30 +/- 35.83	**12815.43 +/- 36.79**	12814.18 +/- 41.23	✓
sp_02	11984.81 +/- 37.01	**12485.71 +/- 39.42**	12491.48 +/- 37.24	✓
sp_03	12220.70 +/- 48.15	**12956.91 +/- 25.10**	12958.25 +/- 32.96	✓
mg_01	11652.49 +/- 43.49	**12470.17 +/- 20.98**	12473.36 +/- 23.95	✓
mg_02	13934.59 +/- 32.24	**15290.95 +/- 38.24**	15297.85 +/- 34.03	✓
mg_03	11259.11 +/- 37.45	**11934.35 +/- 27.87**	11938.57 +/- 25.61	✓
sm_eq_01	12349.25 +/- 69.52	**13117.50 +/- 21.85**	13121.10 +/- 23.80	✓
sm_eq_02	12440.41 +/- 54.46	**13144.34 +/- 26.44**	13143.07 +/- 31.32	✓
sm_eq_03	12841.29 +/- 61.06	**13497.83 +/- 53.04**	13500.51 +/- 51.07	✓
sm_m+_01	12452.29 +/- 40.35	**13090.11 +/- 34.70**	13097.42 +/- 34.68	✓
sm_m+_02	13168.50 +/- 69.71	**14222.04 +/- 54.70**	14223.15 +/- 62.60	✓
sm_m+_03	11767.94 +/- 40.08	**12370.63 +/- 27.91**	12382.08 +/- 28.14	✓
sm_s+_01	11780.35 +/- 33.68	**12441.00 +/- 29.73**	12440.00 +/- 27.17	✓
sm_s+_02	11143.76 +/- 35.95	**11737.89 +/- 21.14**	11739.53 +/- 21.51	✓
sm_s+_03	13282.28 +/- 50.27	**14135.52 +/- 40.05**	14133.48 +/- 36.23	✓

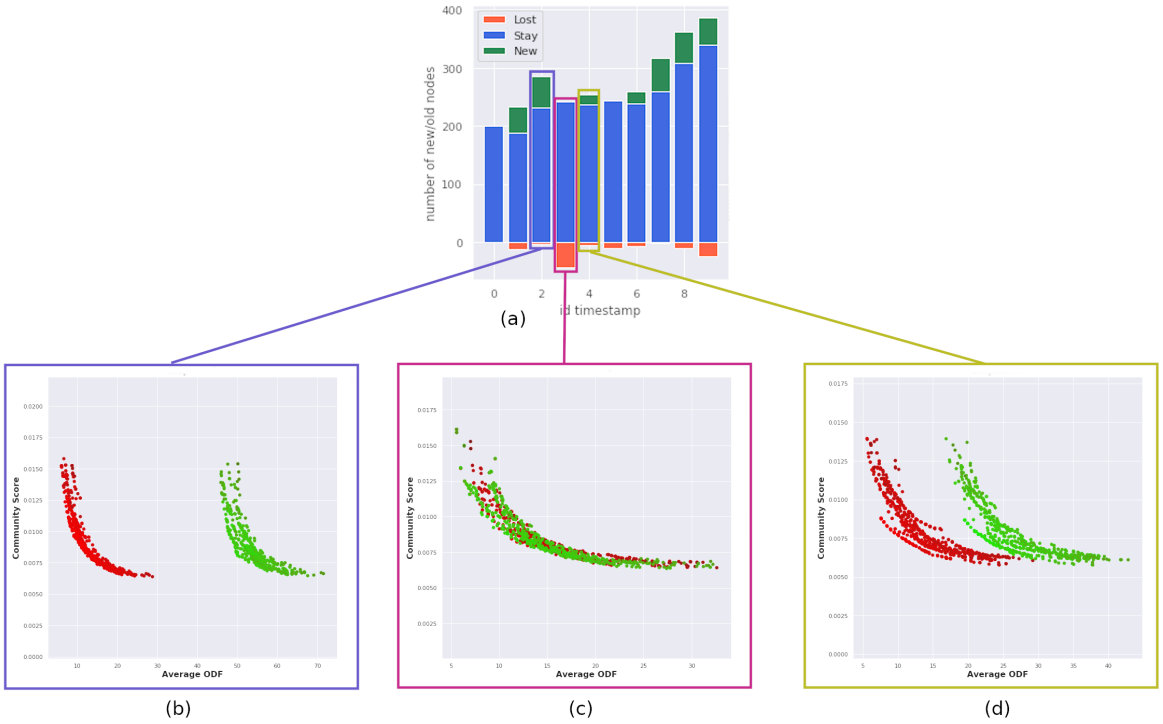


Figure 4: Subfigure (a) contains the difference in nodes between consecutive snapshots of the dynamic network mx_03. Subfigures (b), (c) and (d) show the Pareto front of snapshots 2, 3 and 4 of dataset mx_03 respectively. The green points correspond to the results of the Label Propagation method and the red points to the results of the Random Walks one.

dataset and each reparation technique applied. These tables follow the same nomenclature as Tables 3 and 4. Analyzing Table 5 we can see that in most of the cases the Modularity values obtained are the same for the three methods. Besides, checking the cases where the results are different we can see that the *Hybrid* method gives very similar values compare to the *Label Propagation* one. In addition, we can see that the *Label Propagation* method gives better results than the *Random Walks* one. Nevertheless, the difference between the two is quite small.

Similar results are obtained when analyzing Table 6. However, some small differences are present. First, this time the *Random Walks* method gives better performance than the *Label Propagation* one in some datasets. Second, there are cases where the *Hybrid* method performs better than the other two, like in *mg_01*, *sm_m+_01*, or *sm_m+_02*. Nonetheless, just like Table 5 the difference in performance is small.

In conclusion, given the results obtained from these tests, we have selected the **Random Walks** technique as the best reparation technique. Although it is true that in some cases the Modularity performance of the Label Propagation technique is better, in our opinion, it is not high enough to justify the loss of Hypervolume performance. Regarding the Hybrid technique, in our opinion, the Modularity boost achieved is so small and infrequent compared with the leap of complexity needed to execute it, please note that both methods (Label Propagation and Random Walks) must be executed, that using it is not justified.

5 Experimental Results

In this section, we will test the performance of our method. First, the immigrant's technique will be tested alone by comparing it against the *Standard method*, the same method but without the immigrant's technique. After that, we will compare our method against several other methods from the literature using both synthetic and real-world networks. Due to the stochasticity of our algorithm, each experiment has been repeated 20 times and a Kruskal-Wallis H-test [23] has been used again.

5.1 Immigrant scheme performance

Before comparing our method against the state of the art, we will test its performance using a subset of the synthetic benchmarks introduced in Section 4.1. This particular subset of benchmarks has been selected to include the datasets that do all the micro/macro operations allowed by the benchmark at the same time and in a balanced way. In our opinion, the datasets that perform all types of operations at the same time resembles more a real-world network than the ones that only do one type of operation. Hence, we have considered them more suitable for testing the complete algorithm.

The main purpose of the immigrant's scheme is to reduce the computational time required to process a dynamic network. Therefore, the first test will compare the number of generations needed to process a dynamic network. Then, the quality of the solutions found will be compared in order to assure that the immigrant's technique is not detrimental. In order to assess the quality of a solution, the NMI [9] metric will be used to compare the output of a particular method against the ground truth of a benchmark.

Figure 5 shows the mean number of generations taken (left), and the time needed (right) to process a whole dynamic network. The blue line shows the results for the Standard method and the orange line the results for the immigrant's one. The vertical lines show the minimum and maximum values achieved in the 20 repetitions. As we can see, using the immigrant's scheme drastically reduces the number of generations taken and the time needed to process a dynamic network in all the benchmarks. The difference is so pronounced that visual analysis is enough to assure that the results are different, no statistical test is needed. The improvement is around 53% fewer computation needed to process a dynamic network, with a maximum peak of 70% and a minimum of 30%.

Regarding the quality of the solutions, Table 7 shows the average square error of all the snapshots of a network obtained by each method. The error of a snapshot is calculated as $1 - NMI$. Each cell in the table shows the median and standard deviation of the 20 repetitions. In addition, a $\approx$ symbol indicates that the Kruskal-Wallis H-test gave a p_value higher than 0.05. Contrarily, a triangle indicates otherwise. Finally, if the triangle is green and points up means that the method improves the standard. Otherwise, a triangle will be red and pointing down.

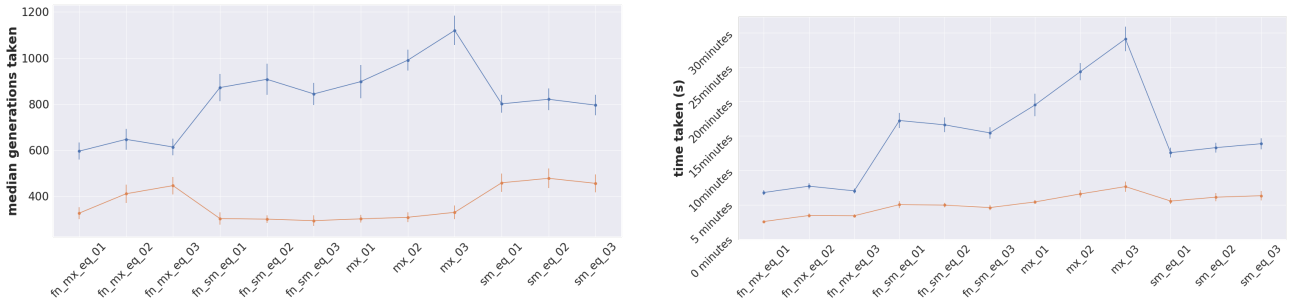
Analyzing the table we can see that the immigrant's technique gives solutions of equal or better quality than the Standard for all the benchmarks except one. Note that the lower the average square error the better a solution is. It is true that the improvement is small but we need to consider that the main purpose of the immigrant's scheme is to reduce computational effort not to improve the quality of the results. However, we can also observe that the standard deviation of the Immigrant's scheme is higher than the Standard's. This is not a positive effect because makes the method less predictable and should be addressed in the future. Besides, the dataset that gives worse results than the Standard not only is the dataset with the lowest mean square error of all, also, is the second dataset with the lowest difference in performance compared with the Standard with a value of 0.012, being the smallest difference 0.008. Nevertheless, the results are positive due to the immigrant's technique drastically reduces the number of generations needed

Table 5: Modularity performance of reparation operators for datasets with a fixed number of nodes. The tick indicates that there is statistical evidence between the results to be considered different, the cross otherwise. Each cell shows the mean value and the standard deviation from 20 repetitions.

Dataset	Label Propagation	Random Walks	Hybrid	
fn_le_c+_01	4.90 +/- 0.04	4.90 +/- 0.04	4.90 +/- 0.04	✗
fn_le_c+_02	5.56 +/- 0.04	5.54 +/- 0.04	5.57 +/- 0.04	✗
fn_le_c+_03	5.28 +/- 0.04	5.30 +/- 0.04	5.31 +/- 0.04	✗
fn_le_d+_01	**3.75 +/- 0.04**	3.68 +/- 0.04	**3.75 +/- 0.04**	✓
fn_le_d+_02	4.34 +/- 0.03	4.31 +/- 0.04	4.35 +/- 0.03	✗
fn_le_d+_03	3.65 +/- 0.03	3.63 +/- 0.03	3.66 +/- 0.03	✗
fn_le_eq_01	**4.97 +/- 0.03**	4.90 +/- 0.03	**4.97 +/- 0.03**	✓
fn_le_eq_02	5.82 +/- 0.05	5.68 +/- 0.08	**5.83 +/- 0.05**	✓
fn_le_eq_03	5.10 +/- 0.03	5.07 +/- 0.04	5.11 +/- 0.03	✗
fn_ge_c+_01	3.61 +/- 0.04	3.61 +/- 0.04	3.61 +/- 0.04	✗
fn_ge_c+_02	4.58 +/- 0.04	4.58 +/- 0.04	4.58 +/- 0.04	✗
fn_ge_c+_03	4.61 +/- 0.05	4.61 +/- 0.05	4.61 +/- 0.05	✗
fn_ge_d+_01	4.71 +/- 0.02	4.71 +/- 0.02	4.71 +/- 0.02	✗
fn_ge_d+_02	3.85 +/- 0.03	3.85 +/- 0.03	3.85 +/- 0.03	✗
fn_ge_d+_03	3.50 +/- 0.04	3.50 +/- 0.04	3.50 +/- 0.04	✗
fn_ge_eq_01	4.31 +/- 0.04	4.31 +/- 0.04	4.31 +/- 0.04	✗
fn_ge_eq_02	3.89 +/- 0.04	3.89 +/- 0.04	3.89 +/- 0.04	✗
fn_ge_eq_03	2.99 +/- 0.06	2.99 +/- 0.06	2.99 +/- 0.06	✗
fn_mx_c+_01	5.84 +/- 0.02	5.84 +/- 0.03	5.85 +/- 0.02	✗
fn_mx_c+_02	4.47 +/- 0.05	4.40 +/- 0.05	**4.48 +/- 0.05**	✓
fn_mx_c+_03	3.84 +/- 0.06	3.76 +/- 0.07	**3.85 +/- 0.05**	✓
fn_mx_d+_01	4.02 +/- 0.02	4.01 +/- 0.02	4.02 +/- 0.02	✗
fn_mx_d+_02	4.14 +/- 0.05	4.11 +/- 0.06	4.15 +/- 0.05	✗
fn_mx_d+_03	**3.97 +/- 0.06**	3.87 +/- 0.05	**3.97 +/- 0.06**	✓
fn_mx_eq_01	4.47 +/- 0.04	4.43 +/- 0.05	**4.48 +/- 0.04**	✓
fn_mx_eq_02	3.80 +/- 0.04	3.78 +/- 0.04	3.83 +/- 0.04	✗
fn_mx_eq_03	**4.98 +/- 0.08**	4.86 +/- 0.07	4.97 +/- 0.07	✓
fn_sp_01	4.66 +/- 0.03	4.66 +/- 0.03	4.66 +/- 0.03	✗
fn_sp_02	5.16 +/- 0.03	5.15 +/- 0.03	5.16 +/- 0.03	✗
fn_sp_03	5.17 +/- 0.03	5.15 +/- 0.03	5.17 +/- 0.03	✗
fn_mg_01	4.70 +/- 0.05	4.70 +/- 0.05	4.70 +/- 0.05	✗
fn_mg_02	4.16 +/- 0.03	4.16 +/- 0.03	4.16 +/- 0.03	✗
fn_mg_03	4.59 +/- 0.03	4.59 +/- 0.03	4.59 +/- 0.03	✗
fn_sm_eq_01	4.34 +/- 0.07	4.33 +/- 0.07	4.34 +/- 0.07	✗
fn_sm_eq_02	3.88 +/- 0.05	3.88 +/- 0.05	3.88 +/- 0.05	✗
fn_sm_eq_03	4.34 +/- 0.05	4.34 +/- 0.05	4.34 +/- 0.05	✗
fn_sm_m+_01	5.82 +/- 0.04	5.82 +/- 0.04	5.83 +/- 0.04	✗
fn_sm_m+_02	3.95 +/- 0.02	3.95 +/- 0.02	3.95 +/- 0.02	✗
fn_sm_m+_03	4.20 +/- 0.05	4.19 +/- 0.05	4.20 +/- 0.05	✗
fn_sm_s+_01	4.19 +/- 0.02	4.19 +/- 0.02	4.19 +/- 0.02	✗
fn_sm_s+_02	5.37 +/- 0.03	5.37 +/- 0.03	5.37 +/- 0.03	✗
fn_sm_s+_03	4.61 +/- 0.03	4.59 +/- 0.03	4.62 +/- 0.03	✗

Table 6: Modularity performance of reparation operators for datasets with a variable number of nodes. The tick indicates that there is statistical evidence between the results to be considered different, the cross otherwise. Each cell shows the mean value and the standard deviation from 20 repetitions.

Dataset	Label Propagation	Random Walks	Hybrid	
ln_01	4.45 +/- 0.08	4.44 +/- 0.07	4.45 +/- 0.08	✗
ln_02	3.48 +/- 0.04	3.48 +/- 0.04	3.48 +/- 0.04	✗
ln_03	4.10 +/- 0.05	4.09 +/- 0.05	4.10 +/- 0.05	✗
gn_01	4.15 +/- 0.04	**4.23 +/- 0.04**	**4.23 +/- 0.03**	✓
gn_02	3.55 +/- 0.04	**3.67 +/- 0.04**	**3.67 +/- 0.04**	✓
gn_03	3.86 +/- 0.06	**3.94 +/- 0.05**	**3.95 +/- 0.06**	✓
mx_01	3.67 +/- 0.06	3.70 +/- 0.06	3.71 +/- 0.06	✗
mx_02	3.10 +/- 0.06	**3.18 +/- 0.05**	**3.18 +/- 0.05**	✓
mx_03	2.90 +/- 0.05	3.07 +/- 0.04	**3.08 +/- 0.04**	✓
sp_01	**4.64 +/- 0.05**	4.57 +/- 0.05	4.62 +/- 0.05	✓
sp_02	**4.45 +/- 0.07**	4.29 +/- 0.07	4.43 +/- 0.08	✓
sp_03	4.13 +/- 0.06	4.07 +/- 0.03	**4.14 +/- 0.05**	✓
mg_01	3.74 +/- 0.04	3.79 +/- 0.04	**3.83 +/- 0.03**	✓
mg_02	3.73 +/- 0.03	3.82 +/- 0.05	**3.85 +/- 0.04**	✓
mg_03	2.75 +/- 0.03	2.82 +/- 0.04	**2.84 +/- 0.04**	✓
sm_eq_01	4.09 +/- 0.07	4.05 +/- 0.06	4.09 +/- 0.06	✗
sm_eq_02	3.20 +/- 0.05	3.23 +/- 0.03	**3.25 +/- 0.03**	✓
sm_eq_03	**3.56 +/- 0.05**	3.50 +/- 0.06	**3.56 +/- 0.06**	✓
sm_m+_01	4.26 +/- 0.06	4.21 +/- 0.06	**4.30 +/- 0.05**	✓
sm_m+_02	3.55 +/- 0.05	3.60 +/- 0.05	**3.64 +/- 0.05**	✓
sm_m+_03	**4.16 +/- 0.06**	4.00 +/- 0.08	4.14 +/- 0.06	✓
sm_s+_01	**4.76 +/- 0.06**	4.67 +/- 0.04	**4.76 +/- 0.04**	✓
sm_s+_02	**4.35 +/- 0.04**	4.24 +/- 0.08	**4.35 +/- 0.04**	✓
sm_s+_03	3.96 +/- 0.04	3.93 +/- 0.04	3.97 +/- 0.03	✗



(a) (b)

Figure 5: On the Left, the mean number of generations taken are shown, and, on the right, the execution time. The blue line is for the Standard method, the orange line is for the method with the immigrant's scheme. The vertical lines indicate the minimum and maximum values.

Table 7: Average mean square error (1-NMI). Each cell shows the mean value and the standard deviation from 20 repetitions.

Datasets	Standard	Immigrant's
fn_mx_eq_01	0.1036 +/- 0.0071	▲ 0.0851 +/- 0.013
fn_mx_eq_02	0.1990 +/- 0.0144	≈ 0.1913 +/- 0.0247
fn_mx_eq_03	0.0550 +/- 0.0083	▼ 0.0671 +/- 0.0104
fn_sm_eq_01	0.0899 +/- 0.0068	≈ 0.0854 +/- 0.0209
fn_sm_eq_02	0.2565 +/- 0.0094	≈ 0.2617 +/- 0.0266
fn_sm_eq_03	0.0908 +/- 0.0064	≈ 0.0966 +/- 0.0161
mx_01	0.2900 +/- 0.0140	▲ 0.2677 +/- 0.0266
mx_02	0.4034 +/- 0.0139	≈ 0.4000 +/- 0.0315
mx_03	0.4466 +/- 0.0104	▲ 0.4275 +/- 0.0321
sm_eq_01	0.1281 +/- 0.0040	▲ 0.1168 +/- 0.0153
sm_eq_02	0.1632 +/- 0.0120	≈ 0.1653 +/- 0.0122
sm_eq_03	0.1685 +/- 0.0143	≈ 0.1640 +/- 0.0165

to process a dynamic network without affecting the quality of a solution to a great extent.

5.2 Performance against the state of art

In this section, we will compare our method against another three methods from the state of the art named: DYNMOGA [14], ALPA [19] and INFOMAP [41]. The first method is also a Multi-Objective Evolutionary Algorithm. Follows the second approach presented in Section 2, just like our method. It has been selected for its similarity with our method in computation complexity and design. The second method works with temporal networks instead of working with a sequence of snapshots. Follows the fourth approach presented in Section 2. It has been selected for its low computation complexity. Finally, the third method is a very common method used for static community detection. In order to compare these methods, we will use both synthetic and real-world datasets. As synthetic datasets, we will use the same benchmarks selected in Section 5.1. On the other hand, as real-world networks, we will use two datasets extracted from the online game Travian [18]. In both datasets, each node of the network represents a player and an edge represents an interaction between two players. Depending on the type of interaction, sending a message or trading, a different dataset is generated. On both datasets, each snapshot in the network contains all the interaction within a day and the whole dynamic network contains data for a complete month. Finally, is worth mentioning that these two datasets came with ground truth extracted from the guild alliances of the players of the game.

Table 8 shows the average square error obtained for all the synthetic datasets presented in Section 2. Each cell of the table contains the mean average square error and standard deviation obtained from the 20 repetitions. The last column of the table indicates with a tick (✓) if the Kruskal-Wallis H-test has given a p_value lower than 0.05 and with a cross (✗) if otherwise. In the column of DYNMOGA, the cells with a "nan" value indicate that the DYNMOGA algorithm was not compatible with the characteristics of that particular dataset. In general, we can see that the values obtained by the four methods are small in general, except in datasets mx_01, mx_02, mx_03 which is a bit higher than the rest. Comparing the method, DYNMOGA works better than the rest on the dataset that could be executed. On the rest of datasets, INFOMAPS works better than ALPA in more cases but ALPA still wins in three datasets. The proposed method obtains the worst results, nevertheless, the difference in performance is low, at around 0.05. Also, our method gives a higher standard deviation compared to others, this behavior was already addressed at Section 2.

Regarding the real-world datasets, Figures 6 and 7 display the difference in nodes and edges between consecutive snapshots of each of the Travian datasets. Both figures are composed of two subfigures. The subfigure (a) shows the difference in the number of nodes and the subfigure (b) in edges. In both subfigures, the X-axis contains the number of nodes/edges and the Y-axis the snapshots. The green section of the bar indicates the number of new nodes/edges compared with the previous snapshot. On the contrary, the red part indicates the number of nodes/edges that have disappeared from the previous snapshot. Finally, the blue part indicates the number of nodes/edges that were present in the previous snapshot as well as in the actual one. Comparing both datasets we can see that the Travian Messages is less dynamic than the Travian Market. The former has less variability in nodes and edges than the latter for all the snapshots. This is especially severe in the case of edges, note that the Market dataset losses most of its edges from one snapshot to the next. Therefore, we can conclude that processing the Market dataset is far more complex than processing the Messages one.

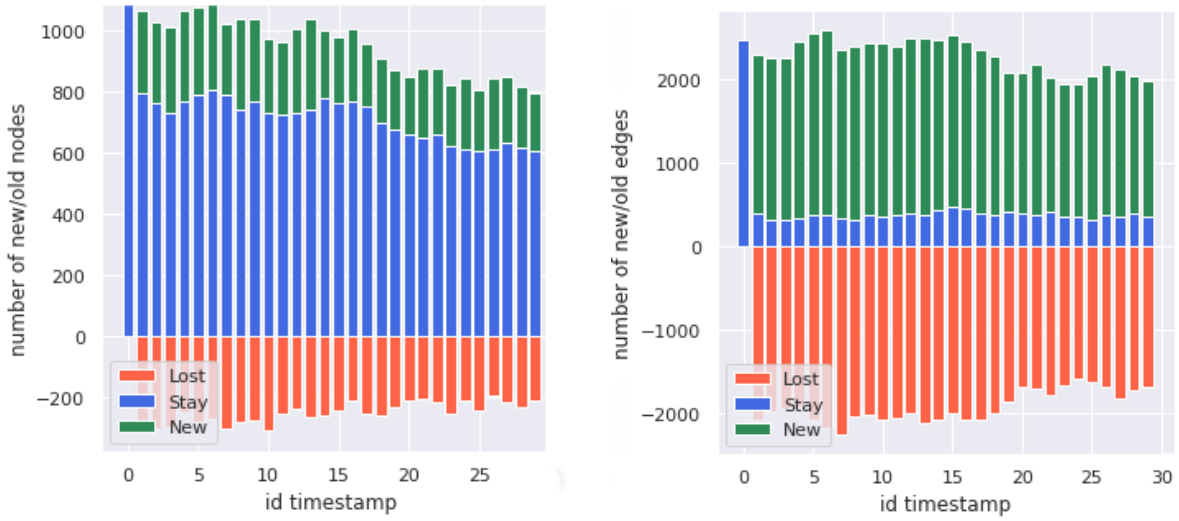
Tables 9 and 10 show the results obtained for the Messages and Market datasets respectively. Conversely with the results shown before, these tables do not show the average square error for all the snapshots, instead, show the NMI values obtained for each snapshot directly. We have done this change because in the tables above our intention was to compare the methods in a plethora of different situation. For this purpose, we have condensed the results of a whole network into a single value, the average square error. In this case, we are interested in analyzing the behavior on two real-world examples giving a fine-grained analysis. Hence, the decision of showing directly the NMI value for each snapshot. As in previous cases, each cell shows the mean and the standard deviation obtained for the 20 repetitions. Also, the last column indicates with a tick (✓) if

Table 8: Average mean square error (1-NMI). Each cell shows the mean value and the standard deviation from 20 repetitions.

Datasets	DYNMOGA	ALPA	INFOMAP	Immigrants	
fn_mx_eq_01	**0.0761 +/- 0.0062**	0.1555 +/- 0.0000	0.0978 +/- 0.0044	0.0851 +/- 0.0130	✓
fn_mx_eq_02	**0.1551 +/- 0.0127**	0.1714 +/- 0.0000	0.1660 +/- 0.0049	0.1913 +/- 0.0247	✓
fn_mx_eq_03	**0.0294 +/- 0.0037**	0.0849 +/- 0.0000	0.0431 +/- 0.0009	0.0671 +/- 0.0104	✓
fn_sm_eq_01	nan +/- nan	**0.0252 +/- 0.0000**	0.0296 +/- 0.0003	0.0854 +/- 0.0209	✓
fn_sm_eq_02	nan +/- nan	0.2089 +/- 0.0000	**0.1812 +/- 0.0025**	0.2617 +/- 0.0266	✓
fn_sm_eq_03	nan +/- nan	**0.0409 +/- 0.0000**	0.0529 +/- 0.0016	0.0966 +/- 0.0161	✓
mx_01	nan +/- nan	0.2126 +/- 0.0000	**0.1881 +/- 0.0024**	0.2677 +/- 0.0266	✓
mx_02	nan +/- nan	**0.2836 +/- 0.0000**	0.3001 +/- 0.0053	0.4000 +/- 0.0315	✓
mx_03	nan +/- nan	0.3800 +/- 0.0000	**0.3507 +/- 0.0046**	0.4275 +/- 0.0321	✓
sm_eq_01	nan +/- nan	0.0831 +/- 0.0000	**0.0794 +/- 0.0020**	0.1168 +/- 0.0153	✓
sm_eq_02	nan +/- nan	0.1292 +/- 0.0000	**0.0746 +/- 0.0030**	0.1653 +/- 0.0122	✓
sm_eq_03	nan +/- nan	0.1136 +/- 0.0000	**0.0935 +/- 0.0026**	0.1640 +/- 0.0165	✓

(a) (b)

Figure 6: Difference in nodes, edges, and neighbors between consecutive snapshots of the Travian Messages datasets. In subfigures (a) and (b), the X-axis shows the difference in nodes/edges and the Y-axis the number of snapshots.



(a) (b)

Figure 7: Difference in nodes, edges, and neighbors between consecutive snapshots of the Travian Market datasets. In subfigures (a) and (b), the X-axis shows the difference in nodes/edges and the Y-axis the number of snapshots.

the Kruskal-Wallis H-test has given a p_value lower than 0.05 or indicates otherwise with a cross (✗). As we can see, the results obtained for the Messages dataset are considerably higher than the ones obtained for the Market one. In the Messages datasets, the values are around 0.9, where the values of the Market one are around 0.5. This is as expected because the Messages dataset representing a dynamic network that is less complex and much easier to process than the Market one.

On the one hand, analyzing the results for the Messages datasets, we can see that our approach obtains worse performance than the other two methods. Our method achieves an NMI around 0.85, where the other two achieve values around 0.9. Between the two other methods, INFOMAP works better in all the snapshot except for snapshot 28 and the difference in performance between both of them is very small, around 0.02. Furthermore, we can see that in all the methods the results are mainly stable and no spikes in performance are present. This is desirable as having a method that works well in some snapshots and poorly in others is troublesome and could generate noise in the results. On the other hand, analyzing the results for the Market dataset, we can see that our approach gives slightly better performance than the other two methods. The output given by our method is around 0.6 while the output of ALPA and INFOMAP is around 0.4 and 0.53 respectively. Moreover, while our method and INFOMAP are fairly stable, ALPA is giving spikes in performance going between 0.25 and 0.5 in a chaotic way.

Before concluding this section, it must be noted that the aim of this article is not to determine when a Multi-Object evolutionary approach is suitable and when not. The aim of this article is to evaluate whether the Immigrant's scheme proposed is a valuable addition to this kind of algorithms. Also, we wish to emphasize that the Immigrant's scheme motif is to reduce computational effort without sacrificing solutions quality. Bearing this in mind, we can conclude that the immigrant's scheme proposed fulfills its purpose. The experiments performed shows that this technique is capable of drastically reducing the generations needed to process a dynamic network, around a 50% less, giving the same performance as if the technique was not being used. In our opinion, this is a very desirable feature as one of the main drawbacks of evolutionary algorithms is their high running times and the Immigrant's technique could make them more competitive in real-world environments. Furthermore, when compared with other methods from the state of art we have obtained similar performance and stability.

However, as aforementioned, evolutionary algorithms are quite slow compared to other kinds of methods. The time that an evolutionary algorithm needs to process a dynamic network is orders of magnitude higher than the time needed by the non-evolutionary methods (ALPA and INFOMAP). If the evolutionary algorithm requires several minutes to process a dynamic network the two other non-evolutionary methods required several seconds. It is also worth mentioning that the implementation used in this work was not made with performance in mind. A lot of cod-

Table 9: NMI values obtained on each snapshot of the Travian messages dataset. Each cell shows the mean value and the standard deviation from 20 repetitions.

	Immigrants	**ALPA**	**INFOMAP**	
Average mean square error	0.0233 +/- 0.0004	0.0110 +/- 0.0000	**0.0081 +/- 0.0000**	✓
Snapshot 0	0.8351 +/- 0.0058	0.8805 +/- 0.0000	**0.9022 +/- 0.0022**	✓
Snapshot 1	0.8490 +/- 0.0039	0.8892 +/- 0.0000	**0.9155 +/- 0.0006**	✓
Snapshot 2	0.8253 +/- 0.0046	0.8681 +/- 0.0000	**0.8903 +/- 0.0014**	✓
Snapshot 3	0.8414 +/- 0.0050	0.8774 +/- 0.0000	**0.8885 +/- 0.0009**	✓
Snapshot 4	0.8440 +/- 0.0045	0.9003 +/- 0.0000	**0.9089 +/- 0.0003**	✓
Snapshot 5	0.8464 +/- 0.0036	0.9087 +/- 0.0000	**0.9210 +/- 0.0010**	✓
Snapshot 6	0.8467 +/- 0.0031	0.9005 +/- 0.0000	**0.9199 +/- 0.0012**	✓
Snapshot 7	0.8424 +/- 0.0035	0.8936 +/- 0.0000	**0.9104 +/- 0.0014**	✓
Snapshot 8	0.8478 +/- 0.0040	0.8970 +/- 0.0000	**0.9221 +/- 0.0011**	✓
Snapshot 9	0.8326 +/- 0.0050	0.8664 +/- 0.0000	**0.8912 +/- 0.0021**	✓
Snapshot 10	0.8365 +/- 0.0044	0.8857 +/- 0.0000	**0.8933 +/- 0.0013**	✓
Snapshot 11	0.8547 +/- 0.0031	0.9037 +/- 0.0000	**0.9335 +/- 0.0009**	✓
Snapshot 12	0.8402 +/- 0.0042	0.8975 +/- 0.0000	**0.9011 +/- 0.0012**	✓
Snapshot 13	0.8266 +/- 0.0029	0.8839 +/- 0.0000	**0.8982 +/- 0.0009**	✓
Snapshot 14	0.8527 +/- 0.0040	0.9123 +/- 0.0000	**0.9383 +/- 0.0011**	✓
Snapshot 15	0.8670 +/- 0.0047	0.9277 +/- 0.0000	**0.9451 +/- 0.0016**	✓
Snapshot 16	0.8366 +/- 0.0041	0.8978 +/- 0.0000	**0.9021 +/- 0.0018**	✓
Snapshot 17	0.8489 +/- 0.0031	0.9211 +/- 0.0000	**0.9371 +/- 0.0011**	✓
Snapshot 18	0.8573 +/- 0.0030	0.9086 +/- 0.0000	**0.9175 +/- 0.0011**	✓
Snapshot 19	0.8401 +/- 0.0041	0.8733 +/- 0.0000	**0.9063 +/- 0.0019**	✓
Snapshot 20	0.8468 +/- 0.0049	0.9212 +/- 0.0000	**0.9271 +/- 0.0012**	✓
Snapshot 21	0.8618 +/- 0.0062	0.9270 +/- 0.0000	**0.9341 +/- 0.0015**	✓
Snapshot 22	0.8752 +/- 0.0045	0.9144 +/- 0.0000	**0.9314 +/- 0.0008**	✓
Snapshot 23	0.8693 +/- 0.0057	0.9082 +/- 0.0000	**0.9292 +/- 0.0016**	✓
Snapshot 24	0.8725 +/- 0.0025	0.9058 +/- 0.0000	**0.9127 +/- 0.0009**	✓
Snapshot 25	0.8536 +/- 0.0042	0.8663 +/- 0.0000	**0.8809 +/- 0.0012**	✓
Snapshot 26	0.8510 +/- 0.0037	0.8883 +/- 0.0000	**0.9151 +/- 0.0009**	✓
Snapshot 27	0.8466 +/- 0.0058	0.9032 +/- 0.0000	**0.9052 +/- 0.0017**	✓
Snapshot 28	0.8397 +/- 0.0040	**0.8657 +/- 0.0000**	0.8657 +/- 0.0015	✓
Snapshot 29	0.8708 +/- 0.0032	0.9053 +/- 0.0000	**0.9140 +/- 0.0013**	✓

Table 10: NMI values obtained on each snapshot of the Travian market dataset. Each cell shows the mean value and the standard deviation from 20 repetitions.

	Immigrants	**ALPA**	**INFOMAP**	
Average mean square error	**0.1705 +/- 0.0009**	0.3620 +/- 0.0000	0.2130 +/- 0.0004	✓
Snapshot 0	0.5249 +/- 0.0103	0.2507 +/- 0.0000	**0.5477 +/- 0.0027**	✓
Snapshot 1	**0.5932 +/- 0.0037**	0.5141 +/- 0.0000	0.5554 +/- 0.0031	✓
Snapshot 2	**0.6011 +/- 0.0055**	0.5265 +/- 0.0000	0.5482 +/- 0.0035	✓
Snapshot 3	**0.6057 +/- 0.0039**	0.4952 +/- 0.0000	0.5528 +/- 0.0019	✓
Snapshot 4	**0.5960 +/- 0.0040**	0.4999 +/- 0.0000	0.5497 +/- 0.0040	✓
Snapshot 5	**0.6008 +/- 0.0036**	0.3450 +/- 0.0000	0.5372 +/- 0.0053	✓
Snapshot 6	**0.5940 +/- 0.0041**	0.2827 +/- 0.0000	0.5299 +/- 0.0021	✓
Snapshot 7	**0.5904 +/- 0.0040**	0.4664 +/- 0.0000	0.5373 +/- 0.0034	✓
Snapshot 8	**0.5980 +/- 0.0044**	0.4690 +/- 0.0000	0.5441 +/- 0.0034	✓
Snapshot 9	**0.5896 +/- 0.0033**	0.4568 +/- 0.0000	0.5306 +/- 0.0036	✓
Snapshot 10	**0.5879 +/- 0.0038**	0.2888 +/- 0.0000	0.5317 +/- 0.0029	✓
Snapshot 11	**0.5935 +/- 0.0054**	0.5191 +/- 0.0000	0.5359 +/- 0.0035	✓
Snapshot 12	**0.5938 +/- 0.0038**	0.2668 +/- 0.0000	0.5389 +/- 0.0055	✓
Snapshot 13	**0.6073 +/- 0.0034**	0.5047 +/- 0.0000	0.5461 +/- 0.0040	✓
Snapshot 14	**0.6031 +/- 0.0048**	0.4558 +/- 0.0000	0.5468 +/- 0.0039	✓
Snapshot 15	**0.5861 +/- 0.0042**	0.2659 +/- 0.0000	0.5258 +/- 0.0036	✓
Snapshot 16	**0.5866 +/- 0.0038**	0.2386 +/- 0.0000	0.5242 +/- 0.0039	✓
Snapshot 17	**0.5956 +/- 0.0044**	0.4043 +/- 0.0000	0.5251 +/- 0.0023	✓
Snapshot 18	**0.5882 +/- 0.0040**	0.4509 +/- 0.0000	0.5345 +/- 0.0036	✓
Snapshot 19	**0.5820 +/- 0.0048**	0.4691 +/- 0.0000	0.5237 +/- 0.0029	✓
Snapshot 20	**0.5876 +/- 0.0038**	0.4677 +/- 0.0000	0.5186 +/- 0.0040	✓
Snapshot 21	**0.5881 +/- 0.0056**	0.3687 +/- 0.0000	0.5407 +/- 0.0032	✓
Snapshot 22	**0.5916 +/- 0.0031**	0.5002 +/- 0.0000	0.5441 +/- 0.0053	✓
Snapshot 23	**0.5916 +/- 0.0063**	0.5147 +/- 0.0000	0.5502 +/- 0.0050	✓
Snapshot 24	**0.6075 +/- 0.0056**	0.5074 +/- 0.0000	0.5529 +/- 0.0042	✓
Snapshot 25	**0.6020 +/- 0.0057**	0.2301 +/- 0.0000	0.5371 +/- 0.0059	✓
Snapshot 26	**0.5951 +/- 0.0058**	0.2591 +/- 0.0000	0.5431 +/- 0.0050	✓
Snapshot 27	**0.6066 +/- 0.0045**	0.4811 +/- 0.0000	0.5512 +/- 0.0062	✓
Snapshot 28	**0.5921 +/- 0.0064**	0.2711 +/- 0.0000	0.5272 +/- 0.0041	✓
Snapshot 29	**0.5925 +/- 0.0044**	0.4503 +/- 0.0000	0.5291 +/- 0.0049	✓

ing improvement, especially in parallel execution, could be added to reduce the computation time, but doing this kind of implementation is out of the scope of this work.

6 Conclusions and Future work

In this article, we present a multi-objective genetic algorithm combined with an Immigrant's scheme for detecting dynamic communities. The motif of the Immigrant's scheme is to reuse previously acquired information in order to speed up convergence. The Immigrant's scheme uses a local search operator for this end. We have also proposed three local search operators that have been thoroughly tuned and tested using a plethora of synthetic benchmarks. Finally, our method has been compared against the state of the art using both real-world and synthetic networks to asses the performance of the technique. After analyzing the results obtained, we can conclude that the Immigrant's scheme more than fulfills its purpose. Using this technique gives around a 50% reduction in the number of generations needed to process a dynamic network without compromising the quality of the solution. For reproducibility all the code necessary to repeat the experimentation presented in this work is available on GitHub[3].

Nevertheless, there is an issue left unaddressed, the experiments have shown an increase in standard deviation when using the Immigrant's technique. We hypothesize that this effect is due to the constant reuse of populations. It is known that the more a population converges to a solution during the evolution process the more exploration capabilities losses. Although, our local search methods are stochastic and introduce some diversity into the population, it looks like this diversity is not enough to fully restore the population's exploratory capabilities. This makes the algorithm more prone to get trap in a local minimum. Hence, as a first future step, we need check if our hypothesis is correct and implement a method that reduces the standard deviation of the Immigrant's scheme to values closer to the ones obtained by the algorithm without immigrants.

In addition, as a second future step, we believe that the local search techniques proposed in this article have potential and could be coupled with other bio-inspired metaheuristic used for detecting dynamic communities. It would be interesting to study and test other approaches. For instance, continuously inserting immigrants from near snapshots into the population during the whole evolution process. Finally, as two future steps in a different direction, we think that the multi-objective nature of the method could be further exploited. In our opinion, generating a whole Pareto Front of solutions instead of one unique solution could give an advantage when it comes to tracking the dynamics of a community throughout the evolution of a network. Thus, we hypothesize that coupling a multi-objective method to detect several promising community structures with another method capable of using them to induce the community dynamics could outperform the actual state of the art in that field. Also, following the same line of thought, we found interesting designing methods capable of detecting and tracking communities following a *Community detection by informed iterative static community detection* approach (see Section 2). In our opinion, the main drawback of the algorithms found in the literature that follows this approach is not addressing the community tracking and forcing a fixed number of nodes. Designing s new method that overcomes this hurdle will be interesting for the community.

[3] `https://github.com/panizolledotangel/Dynamic-communitie-MOGA-FGCS.git`

Acknowledgment

This work has been co-funded by the following grants: Spanish Ministry of Science and Education and Competitivity (MINECO) and European Regional Development Fund (FEDER) under grant TIN2017-85727-C4-3-P (DeepBio), and by Comunidad Autonoma de Madrid under grant P2018 / TCS-4566 (CYNAMON).

References

[1] R. Alhajj and J. Rokne. *Encyclopedia of social network analysis and mining*. Springer Publishing Company, Incorporated, 2014.

[2] G. Bello-Orgaz and D. Camacho. Evolutionary clustering algorithm for community detection using graph-based information. In *2014 IEEE Congress on Evolutionary Computation (CEC)*, pages 930–937, July 2014.

[3] G. Bello-Orgaz, J. Hernandez-Castro, and D. Camacho. Detecting discussion communities on vaccination in twitter. *Future Generation Computer Systems*, 66:125–136, 2016.

[4] G. Bello-Orgaz, H. Menéndez, S. Okazaki, and D. Camacho. Combining social-based data mining techniques to extract collective trends from twitter. *Malaysian Journal of Computer Science*, 27(2), 2014.

[5] G. Bello-Orgaz, H. D. Menéndez, and D. Camacho. Adaptive k-means algorithm for overlapped graph clustering. *International journal of neural systems*, 22(05):1250018, 2012.

[6] P. Bródka, S. Saganowski, and P. Kazienko. Ged: the method for group evolution discovery in social networks. *Social Network Analysis and Mining*, 3(1):1–14, 2013.

[7] D. Chakrabarti, R. Kumar, and A. Tomkins. Evolutionary clustering. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 554–560. ACM, 2006.

[8] H. G. Cobb and J. J. Grefenstette. Genetic algorithms for tracking changing environments. Technical report, Naval Research Lab Washington DC, 1993.

[9] L. Danon, A. Diaz-Guilera, J. Duch, and A. Arenas. Comparing community structure identification. *Journal of Statistical Mechanics: Theory and Experiment*, 2005(09):P09008, 2005.

[10] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002.

[11] S. N. Dorogovtsev and J. F. F. Mendes. *Evolution of Networks: From Biological Nets to the Internet and WWW*. Oxford University Press, 2003.

[12] S. S. Elisa. Survey: Graph clustering. *Comput. Sci. Rev.*, 1(1):27–64, Aug. 2007.

[13] G. W. Flake, S. Lawrence, and C. L. Giles. Efficient identification of web communities. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 150–160. ACM, 2000.

[14] F. Folino and C. Pizzuti. An evolutionary multiobjective approach for community discovery in dynamic networks. *IEEE Transactions on Knowledge and Data Engineering*, 26(8):1838–1852, 2014.

[15] A. Gonzalez-Pardo, J. J. Jung, and D. Camacho. Aco-based clustering for ego network analysis. *Future Generation Computer Systems*, 66:160–170, 2017.

[16] D. Greene, D. Doyle, and P. Cunningham. Tracking the evolution of communities in dynamic social networks. In *Advances in social networks analysis and mining (ASONAM), 2010 international conference on*, pages 176–183. IEEE, 2010.

[17] S. Gregory. Finding overlapping communities in networks by label propagation. *New Journal of Physics*, 12(10):103018, 2010.

[18] A. Hajibagheri, G. Sukthankar, K. Lakkaraju, H. Alvari, R. T. Wigand, and N. Agarwal. Using massively multiplayer online game data to analyze the dynamics of social interactions. *Social Interactions in Virtual Worlds: An Interdisciplinary Perspective*, 2018.

[19] J. Han, W. Li, L. Zhao, Z. Su, Y. Zou, and W. Deng. Community detection in dynamic networks via adaptive label propagation. *PloS one*, 12(11):e0188655, 2017.

[20] J. Hopcroft, O. Khan, B. Kulis, and B. Selman. Tracking evolving communities in large linked networks. *Proceedings of the National Academy of Sciences*, 101(suppl 1):5249–5253, 2004.

[21] R. Kannan, S. Vempala, and A. Veta. On clusteringsgood, bad and spectral. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, FOCS '00, pages 367–, Washington, DC, USA, 2000. IEEE Computer Society.

[22] K. Kim, R. I. McKay, and B.-R. Moon. Multiobjective evolutionary algorithms for dynamic social network clustering. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, pages 1179–1186. ACM, 2010.

[23] W. H. Kruskal and W. A. Wallis. Use of ranks in one-criterion variance analysis. *Journal of the American statistical Association*, 47(260):583–621, 1952.

[24] R. Lara-Cabrera, A. Gonzalez-Pardo, M. Barhamgi, and D. Camacho. Extracting radicalisation behavioural patterns from social network data. In *2017 28th International Workshop on Database and Expert Systems Applications (DEXA)*, pages 6–10, Aug 2017.

[25] R. Lara-Cabrera, A. Gonzalez-Pardo, and D. Camacho. Statistical analysis of risk assessment factors and metrics to evaluate radicalisation in twitter. *Future Generation Computer Systems*, 2017.

[26] R. Lara-Cabrera, A. G. Pardo, K. Benouaret, N. Faci, D. Benslimane, and D. Camacho. Measuring the radicalisation risk in social networks. *IEEE Access*, 5:10892–10900, 2017.

[27] C. Largeron, P.-N. Mougel, O. Benyahia, and O. R. Zaïane. Dancer: dynamic attributed networks with community structure generation. *Knowledge and Information Systems*, 53(1):109–151, 2017.

[28] S. Li, Y. Chen, H. Du, and M. W. Feldman. A genetic algorithm with local search strategy for improved detection of community structure. *Complexity*, 15(4):53–60, 2010.

[29] Y.-R. Lin, Y. Chi, S. Zhu, H. Sundaram, and B. L. Tseng. Facetnet: a framework for analyzing communities and their evolutions in dynamic networks. In *Proceedings of the 17th international conference on World Wide Web*, pages 685–694. ACM, 2008.

[30] A. Malm and G. Bichler. Networks of collaborating criminals: Assessing the structural vulnerability of drug markets. *Journal of Research in Crime and Delinquency*, 48(2):271–297, 2011.

[31] S. B. Mathias, V. Rosset, and M. C. Nascimento. Community detection by consensus genetic-based algorithm for directed networks. *Procedia Computer Science*, 96:90–99, 2016.

[32] P. J. Mucha, T. Richardson, K. Macon, M. A. Porter, and J.-P. Onnela. Community structure in time-dependent, multiscale, and multiplex networks. *science*, 328(5980):876–878, 2010.

[33] M. E. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

[34] G. Palla, A.-L. Barabási, and T. Vicsek. Quantifying social group evolution. *Nature*, 446(7136):664, 2007.

[35] A. Panizo, G. Bello-Orgaz, and D. Camacho. A genetic algorithm with local search based on label propagation for detecting dynamic communities. In *International Symposium on Intelligent and Distributed Computing*, pages 319–328. Springer, 2018.

[36] A. Panizo, G. Bello-Orgaz, A. Ortega, and D. Camacho. *Community finding in dynamic networks using a genetic algorithm improved via a hybrid immigrants scheme*, pages 591–598.

[37] Y. Park and M. Song. A genetic algorithm for clustering problems. In *Proceedings of the third annual conference on genetic programming*, volume 1998, pages 568–575, 1998.

[38] C. Pizzuti. Ga-net: A genetic algorithm for community detection in social networks. In *International Conference on Parallel Problem Solving from Nature*, pages 1081–1090. Springer, 2008.

[39] C. Pizzuti. Boosting the detection of modular community structure with genetic algorithms and local search. In *Proceedings of the 27th annual ACM symposium on applied computing*, pages 226–231. ACM, 2012.

[40] U. N. Raghavan, R. Albert, and S. Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical review E*, 76(3):036106, 2007.

[41] M. Rosvall and C. T. Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 105(4):1118–1123, 2008.

[42] A. Sanfeliu and K.-S. Fu. A distance measure between attributed relational graphs for pattern recognition. *IEEE transactions on systems, man, and cybernetics*, (3):353–362, 1983.

[43] C. Shi, P. S. Yu, Z. Yan, Y. Huang, and B. Wang. Comparison and selection of objective functions in multiobjective community detection. *Computational Intelligence*, 30(3):562–582, 2014.

[44] A. Simões and E. Costa. An immune system-based genetic algorithm to deal with dynamic environments: diversity and memory. In *Artificial Neural Nets and Genetic Algorithms*, pages 168–174. Springer, 2003.

[45] J. Sun, C. Faloutsos, S. Papadimitriou, and P. S. Yu. Graphscope: parameter-free mining of large time-evolving graphs. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 687–696. ACM, 2007.

[46] S. Wang, H. Zou, Q. Sun, X. Zhu, and F. Yang. Community detection via improved genetic algorithm in complex network. *Information Technology Journal*, 11(3):384–387, 2012.

[47] J. Xie, B. K. Szymanski, and X. Liu. Slpa: Uncovering overlapping communities in social networks via a speaker-listener interaction dynamic process. In *2011 IEEE 11th International Conference on Data Mining Workshops*, pages 344–349. IEEE, 2011.

[48] T. Yang, Y. Chi, S. Zhu, Y. Gong, and R. Jin. Detecting communities and their evolutions in dynamic social networks—a bayesian approach. *Machine learning*, 82(2):157–189, 2011.

[49] E. Zitzler and L. Thiele. Multiobjective optimization using evolutionary algorithms—a comparative case study. In *International conference on parallel problem solving from nature*, pages 292–301. Springer, 1998.

Publicación R3 (versión sin el formato de la revista)

ADDRESSING EVOLUTIONARY-BASED DYNAMIC PROBLEMS: A NEW METHODOLOGY FOR EVALUATING IMMIGRANTS STRATEGIES IN MOGAS

Ángel Panizo-LLedot
angel.panizo@upm.es
Departamento de Sistemas Informáticos
Universidad Politécnica de Madrid, Spain

Martín Pedemonte
mpedemon@fing.edu.uy
Instituto de Computación
Universidad de la República, Uruguay

Gema Bello-Orgaz
gema.borgaz@upm.es
Departamento de Sistemas Informáticos
Universidad Politécnica de Madrid, Spain

David Camacho
david.camacho@upm.es
Departamento de Sistemas Informáticos
Universidad Politécnica de Madrid, Spain

March 7, 2022

ABSTRACT

Multi-Objective Genetic Algorithms (MOGAs) have been successfully used to address dynamic problems in a wide variety of domains. In these domains, data changes over time, so a non-static analysis is required to obtain feasible solutions. In this type of environments, MOGAs are often time-consuming and require special adaptation to work properly. A number of different techniques have been proposed to adapt MOGAs to dynamic environments for tackling the previous problems such as hypermutation, memory and immigrant schemes or multi-population methods, among others. In particular, immigrant strategies are one of the most commonly used methods, for this reason, this work proposes a new methodology that allows to make a detailed evaluation of their performance when these strategies are used. The proposed methodology works on two levels, a coarse-grain one and a fine-grain one. In the former, an overall evaluation of the different immigrant strategies is made based on three different dimensions: *Quality*, *Stability* and *Speed*. In the latter, a detailed study of the status of the immigrant individuals during the evolution of the algorithm is carried out. This is a very relevant aspect to take into account in order to evaluate whether an immigrant strategy is working properly or not. To deploy this methodology, a new visualization technique for population mixing analysis is proposed in this work. In order to validate the proposed methodology, a test case in the context of the Dynamic Community Detection problem (DCD) has been selected using a MOGA that applies several different immigrant schemes, showing both how the methodology works and how it could be employed in a particular dynamic problem.

***K*eywords** Dynamic problems · Immigrant strategies · Multi-objective genetic algorithms · Dynamic community detection · Social network analysis

1 Introduction

In areas such as computational complexity theory, dynamic problems are defined as a kind of problems where input data changes over time [1]. This simply fact critically affects how the information is represented and how the algorithms handle data. These algorithms, usually known as *dynamic algorithms*, need to manage two main issues, the first one related to the memory space required to store data, and the second one related to a set of time-based features (initialization, insertion, deletion, query, etc...) and how the algorithm uses data in different time stamps [2]. Many real-world problems are dynamic in nature, in particular within the optimization field. Dynamic Multi-objective Optimization Problems (DMOPs) are widely present in real world [3]. This type of problem is characterized by time-varying objective functions in which at least two goals can be pursued. Traditionally, solving DMOPs implies

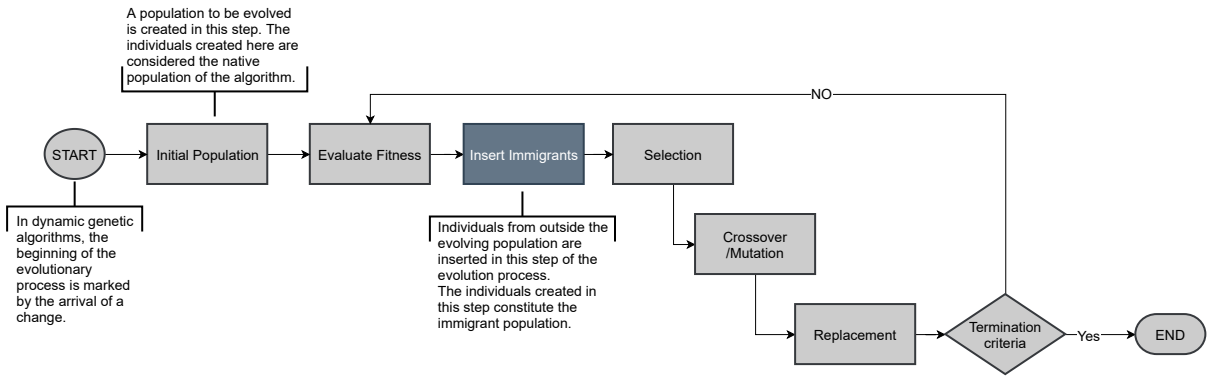


Figure 1: Flow chart of a genetic algorithm that integrates an immigrant strategy in its algorithmic scheme. The steps involved in the vanilla version of the algorithm are shown in light (grey) color. Contrary, the step added for the immigrants variant of the algorithm is shown in a dark one.

not only evolving the solutions close to the optimal Pareto Front (PF) in any given timestamp but also doing it quickly before a change occurs. In recent years, another goal has also been proposed, which implies finding solutions robust to changes over time (ROOT). [4].

DMOP is a very active area of research in evolutionary computation [5, 6], especially in relation to Multi-Objective Genetic Algorithms (MOGAs). To address these problems, MOGAs require that the optimization algorithm can trace the Pareto-optimal over time, increasing its computational complexity. Therefore, MOGAs should be able not only to evolve a near-optimal and diverse PF, but also to continually track a time-changing environment. However, it is unreasonable to assume no priori knowledge whenever a change occurs in the environment. Hence, some extent of knowledge transfer [7]is mandatory when designing MOGAs for dynamic problems. Although more complex techniques exist, such as those proposed in the emerging field of Evolutionary Multi-Task Optimization (EMTO) [8], the reuse of individuals from previous generations is the simplest and most popular technique for sharing knowledge in genetic algorithms for dynamic problems. Researchers in this area have proposed many methods to address this issue [9]. Some of the techniques presented for this purpose are *hypermutation* [10] (increases the mutation rate of the population when a change occurs in the environment), *memory schemes* [11] (maintains a memory of individuals during all the MOGA execution, and when a change occurs, this memory is used to influence the current population), and *immigrant's* [12] (inserts new individuals into the population when a change occurs in order to improve the population's diversity). Among the different approaches mentioned, the *immigrant's* technique is one of the most widespread and popular [10, 13–15]. However, tuning this method for optimal performance is not a straightforward task. Setting the number of immigrants to be inserted, the replacement strategy of those immigrants, or boosting their survival probability is a challenging task [16]. These parameters critically affect the performance of the MOGA, and it is difficult to get insights about what could be malfunctioning. For this reason, this work focuses on proposing a methodology that allows an in-depth analysis of the effectiveness of the application of this method to solve dynamic problems.

Summarizing previous algorithms and methods, any immigrant technique is based on the idea of inserting new individuals from outside the current evolving population to replace a predefined part of that population. The underlying hypothesis is that inserting new individuals will help to improve the convergence speed and/or increase the population diversity (two of the most fundamental and critical features for any evolutionary algorithm). Figure 1, shows the flow chart of a generic genetic algorithm (GA) that incorporates a basic immigrant strategy. Depending on the immigrant technique selected, the *Initial Population* and *Insert Immigrants* steps in the GA will vary. There are several strategies proposed in the literature for this type of technique, some of the most commonly applied are the following:

1. **Random Immigrant Genetic Algorithm** (RIGA) [17]: Algorithm 1 contains a pseudo-code that implements the RIGA method. In this immigrants scheme, the initial population is composed by the solutions of the population that was being evolved previously to the new change (line 1). In the *Insert Immigrants* step (line 11), newly created random individuals are inserted into the population replacing the worse individuals. This technique aims to enhance the exploration capabilities of the algorithm by making the population more diverse.
2. **Elite Immigrants Genetic Algorithm** (EIGA) [18]: Algorithm 2 contains a pseudo-code that implements the EIGA method. In this immigrants scheme, the initial population is a completely new population that is generated randomly (line 1). In addition, the population of the MOGA before the change happens is stored

in an external memory to be used as immigrants. In the *Insert Immigrants* step (line 11), individuals from the external memory are inserted into the population, replacing the worse individuals. This technique aims to enhance the convergence speed of the method by reusing past information (assuming smooth changes in the environment).

3. **Hybrid Immigrants Genetic Algorithm** (HIGA) [19]: Algortihm 3 contains a pseudo-code that implements the HIGA method. This immigrants scheme combines previous techniques depending on the population needs, improving convergence speed or improving the exploration necessities by inserting a random immigrant or an elite one. In this scheme, the initial population is a completely new random population (line 1) and also stores the population before the change in an external memory (as in EIGA). However, contrary to EIGA, this method combines the insertion of immigrants with the *Crossover/Mutation* phase. The method works as follows. If both parents are not members of the non-dominated set of solutions, then an individual from the external memory is inserted (lines 7-8). If at least one of the parents is a member of the non-dominated set and the distance between both parents is less than 10% of genes, then a completely random individual is inserted (lines 10-11). Otherwise, a classical crossover is used (lines 12-13).

Algorithm 1 RIGA

Require: P^{t-1}
1: $P^t \leftarrow updatePrevious(P^{t-1})$
2: $P^t \leftarrow evaluate(P^t)$
3: **while** !$termination_criteria()$ **do**
4: $parents \leftarrow selection(P^t)$
5: $offspring \leftarrow \emptyset$
6: **for all** $p_1, p_2 \in parents$ **do**
7: $child \leftarrow crossover\&mutation(p_1, p_2)$
8: $offspring \leftarrow offspring \cup child$
9: **end for**
10: $P^t \leftarrow replacement(P^t, offspring)$
//N random individuals replace the worse of P^t
11: $P^t \leftarrow insertRandom(P^t, N)$
12: **end while**
13:
14: **return** P^t

Algorithm 2 EIGA

Require: P^{t-1}
1: $P^t \leftarrow createRandom(POP_SIZE)$
2: $P^t \leftarrow evaluate(P^t)$
3: **while** !$termination_criteria()$ **do**
4: $parents \leftarrow selection(P^t)$
5: $offspring \leftarrow \emptyset$
6: **for all** $p_1, p_2 \in parents$ **do**
7: $child \leftarrow crossover\&mutation(p_1, p_2)$
8: $offspring \leftarrow offspring \cup child$
9: **end for**
10: $P^t \leftarrow replacement(P^t, offspring)$
//N previous individuals replace the worse of P^t
11: $P^t \leftarrow insertPrevious(P^t, P^{t-1}, N)$
12: **end while**
13:
14: **return** P^t

The implementation of any particular immigrant strategy and its inclusion in a particular MOGA, in order to enable the algorithm to solve DMOPs, is often quite straightforward. However, when it comes to the empirical analysis to study how the immigrant strategy actually affects the evolution of the population and the numerical performance of the algorithm, the task becomes complex and hard. Therefore, and with the goal of making a step further in this direction, in this work we propose a new methodology that allows studying in detail the efficiency of the application of the immigrant

Algorithm 3 HIGA

Require: P^{t-1}
$P^t \leftarrow createRandom(POP_SIZE)$
$P^t \leftarrow evaluate(P^t)$
while $!termination_criteria()$ **do**
  $parents \leftarrow selection(P^t)$
  $offspring \leftarrow \emptyset$
  for all $p_1, p_2 \in parents$ **do**
    if $\{p_1, p_2\} \cap nonDominate(P^t) == \emptyset$ **then**
      $child \leftarrow selectInd(P^{t-1})$
    else
      if $genDistance(p_1, p_2) \leq 0.1$ **then**
        $child \leftarrow makeRandomInd()$
      else
        $child \leftarrow crossover\&mutation(p_1, p_2)$
      end if
    end if
    $offspring \leftarrow offspring \cup child$
  end for
  $P^t \leftarrow replacement(P^t, offspring)$
end while

return P^t

strategies to MOGAs from a quantitative perspective. Our methodology makes possible to analyze if the optimization algorithm is finding sets of non-dominated solutions close to the PFs, as well as, well-distributed through the front and adapted to changes in the environment over time. When the algorithm fails to fulfill such goals, our methodology allows identifying the underlying causes via a new visualization technique. This technique is inspired by the works proposed to analyze the performance of cellular genetic algorithms [20]. Specifically, the *takeover* time [21] and the solution's flow presented in [22]. On the one hand, the *takeover* time, which measures the time it takes a good solution to dominate a population, serves as a theoretical basis for the visualization technique since its main utility is detecting if the native or immigrant population is dominating the population too quickly. On the other hand, the flow analysis done in [22] inspired the categorization of the individuals depending on their ancestors.

This work is not the first of its kind. During the past two decades, several methods have been proposed to evaluate and compare dynamic genetic algorithms, including the ones using immigrant strategies. It is common to show graphs comparing the mean value of the objective function over time or use adaptations of accuracy metrics to dynamic environments. For example, Yun. et al [16] evaluate dynamic algorithms based on three metrics: the best/average performance, the best/average robustness, and the diversity. Or R.W.Morrison [23] who propose the Mean Fitness metric. Without forgetting the metrics that require knowing the best solution to a problem like the Accuracy and the Adaptability presented in [24] and [25]. It is worth nothing to mention that all the immigrant strategies work on top of genetic algorithms designed for static problems. However, all the methods proposed in the state of the art analyze the immigrant strategies in isolation, without taking into account the base genetic algorithm. The main novelty in our methodology is that it gives priority to the base genetic algorithm. With this methodology is not only possible to assess if an immigrant strategy works better than another but also assess if it improves o hinders the base algorithm in some factor.

In addition, the visualization technique proposed allows getting an insight into the evolution process that other methods cannot. It is common knowledge that random immigrants work best when the changes are "big" and elite immigrants work best if the changes are "medium or small". However, the terms "big", "medium" or "small" are open to debate and highly dependent on the problem. It is difficult to know when a change is big or medium for a particular problem and the visualization technique allows checking if the immigrants are working correctly whatever the size of the changes are. It is true that there are metrics that try to give understanding about the immigrant dynamics during the evolutionary process, like the "immigrant effect" proposed in [26] or the aforementioned "diversity" [16]. But it is also true, that those metrics were not designed with that goal in mind, notice that the goal of the "immigrant effect" is to adapt the immigrant insertion. Which, in our opinion, makes our technique more flexible and powerful.

Therefore, the main contributions of this work can be briefly summarised as follows:

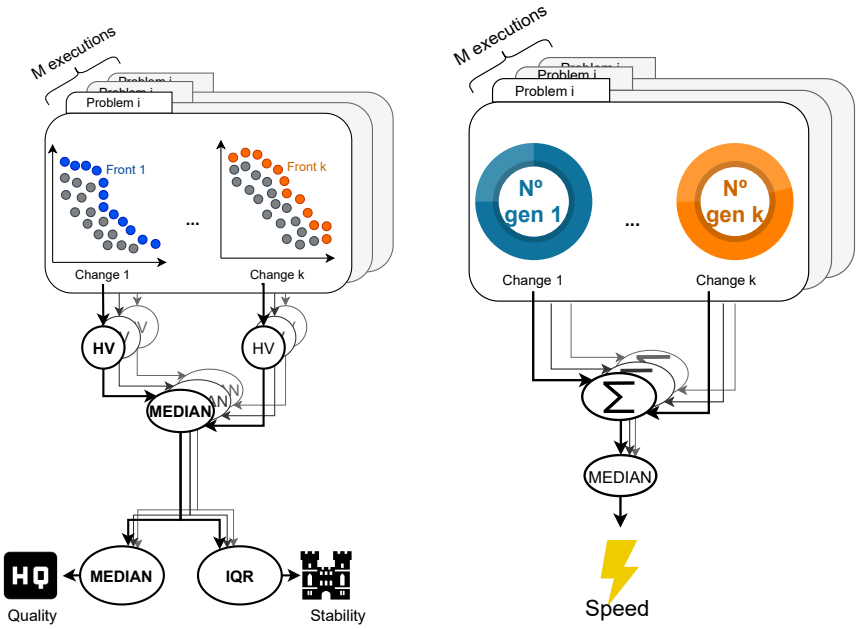


Figure 2: (a) Workflow for assessing the Quality and Stability dimensions. b) Workflow to assess the Speed dimension.

1. A **new methodology** for conducting in-depth and quantitative evaluations of the performance of immigrant strategies is proposed on the basis of three different dimensions: the *quality* of the solutions, the convergence *speed*, and *stability* of the results obtained.

2. A **new visualization analysis technique** for studying how the solutions of the populations are mixed, as well as, the status of the immigrant individuals during the evolution of the MOGA, is presented.

3. An **empirical validation** of our proposed methodology using a real-world test case is conducted using a MOGA that employ different immigrant schemes in a Dynamic Community Detection (DCD) problem.

The reminder of the article is organized as follows. A detailed description of the methodology proposed in this paper is presented in Section II. Section III, introduces the dynamic community detection problem as a case study for validating the methodology presented. This section shows how our proposal can be used for an in-depth analysis of different immigrant schemes while addressing a real-world problem. Finally, Section 4 outlines the main conclusions of this work and the future lines of work.

2 Methodology

The methodology proposed for evaluating the performance of immigrant strategies is based on the usage of three different metrics or dimensions: *Quality*, *Speed*, and *Stability*. The next subsections describe these dimensions, and the workflow followed to assess them. A requirement to apply our methodology is that two or more different immigrant strategies are under study. Since our methodology is proposed for analyzing the performance over several instances, we assume that there are N different instances of the same problem previously selected. For example, N dynamic networks when considering the Dynamic Community Detection Problem or N jobs/machines configurations when working on Job-shop scheduling problems. In addition, to obtain robust statistical evidence, we assume that each problem has been executed independently M times using each of the immigrant strategies selected. It should be noted that N and M should be chosen taking into account the type of problem and the desired statistical significance. Finally, we take for granted that the changes occurring during each of the N problems are a priori known. From these samples, our approach assess the performance of the strategies under study.

2.1 Dimensions

As aforementioned, each immigrant scheme is evaluated using three different dimensions: *Quality*, *Speed* and *Stability*. Each dimension is calculated using the results from the M executions of each instance of the problem independently. Figure 2 shows the workflow for calculating each dimension. These dimensions are described in detail below:

- *Quality*. This dimension measures the quality of the Pareto Fronts generated by an immigrant strategy. One of the most popular metrics for measuring the quality of Pareto Fronts, and the one used in our methodology, is the Hypervolume (HV) [27]. The HV measures the area of the objective space covered by a non-dominated set with respect to a reference point. In order to make a fair comparison of the HV obtained when working with different objective functions and problem instances, a common reference point must be set and each Pareto Front obtained by the algorithm needs to be normalized. This normalization uses the maximum and the minimum values found by all the immigrant strategies in all the problem instances and all the executions. To calculate the Quality metric of a particular instance, first, the median of the HV of the normalized Pareto front generated after each change for each execution of a problem instance is calculated. Next, the median of the M executions is used as the final *Quality* metric value. It should be noted that the higher the value of this metric, the better the performance of the immigrant strategy.
- *Speed*: This dimension measures the convergence speed of an immigrant strategy. This is a crucial aspect, since the whole purpose of the immigrant technique is to reuse previous information (i.e. knowledge) to accelerate the evolutionary process. To measure the Speed metric for an instance of the problem, first, the number of generations required to converge after each environment change is calculated. A method is faster than another method when a fewer number of generations is needed to converge. For this reason, we total the values previously calculated, i.e., we calculate the sum of the number of generations until the algorithm converges after each environment change. Finally, the median of this value over the M repetitions is used as the Speed dimension value.
- *Stability*. This dimension measures the reliability of the results of an immigrant strategy. The application of an immigrant technique could influence the exploration capabilities of the MOGA, making it less or more prone to get stuck in local minima. When the latter happens, the MOGA becomes less reliable, as a single run of the evolutionary algorithm could give a worse result with a higher probability. The procedure to calculate this metric is the same as in the *Quality* metric, except for the last step. For the *Stability* metric, instead of using the median of the M executions of each test case, the Interquartile Range (IQR) of the values is used. The IQR is a measure of the dispersion of the data. Therefore, the higher the value, the lower the stability, causing that the quality of the solutions obtained has a higher variability between different runs.

2.2 Workflow

The workflow of the methodology proposed is composed of three steps. First, all the immigrant strategies under study are compared against the same MOGA without immigrants. From now on, we will refer to this algorithm as *Standard*. Then, in the second step, all the immigrant strategies are compared against each other, in order to analyze which ones present better results. Finally, a fine-grain analysis takes place, where specific problem instances are evaluated independently using the *Population mixture visualization technique*. Next, we detail each of the steps of the workflow proposed.

In the first step, all the immigrant strategies are compared against *Standard* for each dimension independently. To this end, the Friedman Signed Ranked Test and the Holm post-hoc analysis [28] are applied over the scores obtained by each strategy in all the problem instances. This allows discerning whether an immigrant strategy improves or hinders some dimension. An immigrant strategy, to be considered successful, should improve the *Speed* dimension while maintaining a similar or better performance in the *Stability* and *Quality* metrics. Since the main purpose of an immigrant strategy is speeding the convergence when using MOGAs in dynamic environments, we consider mandatory the improvement of the *Speed* dimension. However, applying immigrants techniques should not hinder the exploration capabilities of a MOGA. Therefore, we expect the algorithm to perform at least as good as the base technique (the one that does not use immigrants).

Regarding the visualization of these results, we recommend presenting this information on a table where the rows are the different immigrant strategies and the columns are the three dimensions. Table 5 shows an example of this visualization. Each cell of these columns contains a green triangle pointing up (▲) if the rank obtained by the immigrant strategy is higher (lower in case of minimization) than the one obtained by *Standard* and there is enough statistical evidence to support that the immigrant strategy performs differently from *Standard*. On the contrary, a cell contains a red triangle pointing down (▼) if the rank obtained is lower (higher in case of minimization). Finally, a cell contains the

	i	**i+i**	**n+i**	**n+n**	**n**
i	i+i	i+i	i+i	n+i	n+i
i+i		i+i	i+i	n+i	n+i
n+i			n+i	n+n	n+n
n+n				n+n	n+n
n					n+n

Table 1: The rows and columns show the type of the parents, while the cells depict the type of the resulting offspring after crossover. Since the table is symmetric, only the upper diagonal of the table is shown, for the sake of simplicity.

approximated sign ($\approx$) when there is not enough statistical evidence to consider that the immigrant strategy performs differently from *Standard*.

The previous step allows understanding if a strategy improves or hinders some dimension. However, it does not give any information on the differences of the metric values for each of the algorithms. Hence, in the second step, all the outputs of the different immigrant strategies are compared against each other using a Friedman Signed Ranked Test and a Shaffer post-hoc analysis [28]. Regarding the visualization of this step, we suggest depicting the relation between the results obtained by each of the different strategies using a graph. Figure 3 shows an example of the proposed visualization. In this graph, the nodes represent the different immigrant strategies, and the edges join two nodes when there is not enough statistical evidence to consider that the two methods perform differently. In addition, the size of the nodes is inverse proportional to the rank obtained by each strategy for the quality dimension and directly proportional for the speed and stability ones, i.e., bigger nodes imply a better performance.

Finally, for the strategies that do not work as intended, the methodology includes the third step, where a fine-grain analysis is performed. First, problem instances that present difficulties are identified following the same process as in the first step, except that each problem instance is evaluated independently. It should be noted that that in this step, instead of using the median of the M executions of the different dimensions, the M values are used. Once the problem instances that are hard to solve for the strategy have been identified, it has to be identified in which periods, regarding the time stamps or environment changes, the evolution process is not working properly. This is done using HV evolution (Figure 4 (b)) and Empirical Attainment Surfaces [29] (Figure 4 (c)) plots. On the one hand, the HV evolution allows identifying if a particular strategy is suffering from premature convergence. On the other hand, the Empirical Attainment Surfaces, which graphically shows a boundary between dominated and non-dominated points, allows identifying areas of the Pareto Front that are not being explored enough by the strategy. Finally, insight about what is hindering the evolution process can be obtained using the *Population mixture* visualization technique on those specific periods. This technique helps to discern whether the immigrants of a strategy are contributing to the improvement of the population, are taking over the entire population, or, on the contrary, are being excluded from the population.

2.3 Population mixture visualization technique

This section describes the visualization technique designed to analyze what is going on with the immigrants during the evolution. To this end, each individual of the population is labeled with a particular type. When two individuals mate, their types determine the new type of their offspring. During the evolution, the number of individuals of each type in the population is counted on each generation. Finally, these values are plotted as curves where one axis contains the generation (first, second, third, and so on), while the other axis has the number of individuals in the population of a particular type. In addition, the curve for each type of individual receives a different color. Figure 7 shows several examples of this kind of visualization.

The available label types for an individual are:

(1) *n (native)*: an original individual of the genetic algorithm.

(2) *i (immigrant)*: individuals external to the original population, which are continuously inserted into the population.

(3) *i+i (immigrant's offspring)*: individuals whose parents are two immigrants or immigrant's offspring.

(4) *n+n (native's offspring)*: individuals whose parents are two native or native's offspring.

(5) *n+i (half-blood)*: an individual which one of the parents is a native or native's offspring, while the other parent is an immigrant or immigrant's offspring.

Table 1 shows the five proposed types of immigrants and the type of the resulting offspring after they are generated by mating. For example, when two parents of type *'i'* mate (first row and column of the table), the resulting offspring is of type *'i+i'*. Likewise, if for example, the parents have types *'n+n'* and *'n+i'* (third row and fourth column in the table), the resulting offspring is of type *'n+n'*.

Analyzing the curves that show the evolution of the different types of individuals during the execution, can help to understand whether the immigrants are contributing to the evolution of the population, or are taking over or being excluded from it. Specifically, the presence of a half-blood (n+i) population curve indicates that a healthy mixture of both populations, native and immigrant, is taken place. Hence, the immigrants are contributing to the local population. However, on the one hand, the over-representation of the immigrant's offspring (i+i) population curve indicates that the immigrants are taking over the native population. If this effect is too extreme, all the native population is expelled from the evolutionary process without a chance of passing their genes to the next generations, which is equivalent to evolve the immigrant population alone. On the other hand, the over-representation of native's offspring (n+n) population curve indicates that the immigrants are not contributing to the evolution process. In extreme cases, all the immigrants can be expelled from the population without having a chance to pass their genes to the next generations. This is equivalent to using a smaller population because, even though the immigrants are part of the population, they are not allowed to participate in the evolution process. These two latter cases usually result in a loss of performance by the immigrant strategy.

3 Case study: Dynamic community detection

In order to test the validity of the proposed methodology, a test case is studied. In particular, we applied our methodology for studying a Multi-Objective Genetic Algorithm that combines an Immigrant's scheme with local search strategies to the dynamic community detection problem [30]. In this case, we use the proposed methodology for evaluating and selecting the most suitable immigrant scheme.

The remainder of this section first introduces some preliminaries on the DCD problem.. Then, the problem instances used to conduct all the experiments are introduced. After that, the particularities of the MOGA for solving the DCDP are explained. Finally, the results obtained by applying the proposed methodology are presented and a detailed analysis is performed.

3.1 Basics on Dynamic Community Detection

The Dynamic Community Detection problem is the dynamic version of the Community Detection Problem (CDP), which consists in splitting a network into groups of nodes, usually known as communities. In a nutshell, it could be considered as clustering in graphs. There is no an unique definition of *Community* agreed upon by the scientific community. Nevertheless, a widely accepted one is that nodes belonging to the same community must be strongly interconnected, while maintaining sparse connections to nodes of other communities.

Static Community Detection is a hot research area and a lot of different methods have been proposed to tackle this problem over the years [31–34]. Of all the strategies used to detect communities, bio-inspired meta-heuristics have proven great success [35], specially genetic algorithms both on the mono-objective [36] and multi-objective [37, 38] configurations.

However, as in so many other areas, the dynamic version of the community detection problem has been neglected until recently. The exponential growth of online Social Networks has increased the demand for efficient dynamic community detection algorithms, which has boosted the interest in the area. When considering the time variable, the quantity of possible analyses broadens. When time is involved, one can be interested in finding one community structure that fits the whole network's lifetime (like in ROOT), detecting significant points on the network's timeline, or studying communities' dynamics (if they split, merge, grow ...etc.) among others. Hence, the DCD area is composed of an amalgamation of problems quite different in nature. We refer the readers interested in this topic to the review [39], where both the static and dynamic variants of the problem are discussed in extensive detail and the most common tools available are reviewed.

In this case study, we will center on the problem of finding a suitable community structure for each point of a network's timeline. The most popular framework to do that is the *evolutionary clustering* [40]. This framework aims to imbue the solutions with a certain smoothness. In evolutionary clustering, two forces must be balanced, one that pushes communities to adjust to the present moment and another that constrains them from deviating from their history. Undoubtedly, bio-inspired meta-heuristics, especially MOGAs, are one of the most popular techniques to do evolutionary clustering. F.Folino y C.Pizzuti [41], C.Guoqiang et.all [42], B.A.Attea y K.Haider [43], and M.Jingjing et.all [44] have proposed MOGAs that follow the evolutionary clustering framework. The former two are based on

NSGA-II [45] and the latter two on MOEA/D [46].Outside genetic algorithms, other bio-inspired metaheuristics have also been proposed to a lesser extent. For example, Y.Ying et.all [47] have proposed a Particle Swarm Optimization (PSO) with custom initialization and crossover operators, and Z.Xu et.all [48] have proposed a Bat Algorithm (BA). However, all of the above methods define trivial criteria for selecting a single Pareto front solution, namely, they stick with the one that best fits the current network topology. Which favours one objective over another. How to make such a decision remains an open question in the area that is worth investigating.

Contrary to the examples above, the MOGA used for this case study does not follow the evolutionary clustering framework. Instead, it forgets about the recent history of the communities and focuses on their quality at the present moment. Our MOGA optimizes two different negatively correlated quality metrics, as recommended in [49]. Note that the aim of this use case is not to propose a better MOGA for DCD and that the conclusions drawn for our MOGA could easily be extrapolated to other similar MOGAs in the state of the art.

3.2 Problem formulation

Given a dynamic network modeled as a sequence of snapshots $N = \{G^0...G^T\}$. Where T is the number of timestamps and each element $G^t(V^t, E^t) \wedge 0 \leq t \leq T$ of the sequence is a graph composed of a set of V^t nodes and a set $E^t = \{(u, v)|u \in V^t, v \in V^t\}$ edges. The objective of the community detection problem is to generate a set of dynamic communities $\{DC_0...DC_m|DC_i = \{C_i^t|0 \leq t \leq T\}\}$ where m is the number of dynamic communities and each element of a dynamic community DC_i is in turn a static community $C_i^t = \{u|u \in V^t\}$. However, it is normal to find this problem split in two in the state of the art. Usually, a method finds promising C_i^t communities for each snapshot G^t and a different method builds the dynamic communities DC_i using those promising communities C_i^t. The methods proposed in this case study tackle the first part of the problem. Hence, given a dynamic network $N = \{G^0...G^T\}$, their objective is to find a set of promising crisp static communities $\{C_0^t...C_n^t\} \wedge \bigcup_i = 0_n C_i^t = \emptyset$ for each snapshot G^t.

3.3 Problem instances under study

For testing the performance of the different immigrant techniques, 18 different synthetic networks have been used (the N parameter in our methodology). The benchmark is composed of these dynamic networks and their corresponding ground truth. We have opted for using only synthetic networks because with them we can generate a benchmark where it can be ensured that a complete and balanced set of scenarios take place. To generate the benchmark the DANCER software has been used [50]. This software allows creating dynamic networks, configuring the number of changes that occur during the evolution of a dynamic network, as well as, the type of these changes. Two different operations are available for generating the dynamic networks:

- The *micro* operation introduces little changes in the network at the node scale (insert/delete a few edges or nodes).
- The *macro* operation introduces big changes in the network at the community scale by merging and splitting them.

Taking this into account, synthetic networks with different types of dynamics are generated to study the efficiency of different immigrant strategies for dynamic networks with different characteristics. Therefore, the synthetic networks generated use all the *micro operations* available all the time. However, for the generation of each benchmark a different combination of *macro operations* is used. It is worth mentioning that to generate each dynamic network both operations are applied with different intensities, where a higher intensity implies that more changes occurs during the evolution.

Table 2 shows the different synthetic dynamic networks generated for testing and the specific characteristics of each one to test. Each row of the table corresponds to one combination of macro operators. All the networks contain nine snapshots (i.e., changes in the network) with between 100-200 nodes and 200-1000 edges. The naming system for the different types of networks consists in several letters followed by a '_XX'. The letter indicate the type of operations that happens in the networks while the '_XX' indicates the intensity. The nomenclature for the intensities is 01 for *low*, 02 for *medium* and 03 for *high*.

3.4 MOGA parametrization for the DCDP

The MOGA analyzed for the DCDP uses the NSGA-II algorithm [51] to optimize two objectives: the "Community Score" [52] and the "Average ODF" [53]. The "Community Score" is an inter-community objective, which calculates how strongly the nodes of a community are connected. In particular, it measures the density of connections in relation to the number of nodes inside a community. Equation 1 shows the formula followed to calculate the Community Score.

Name	Description
mx_XX	Network applying *micro operations* during its evolution with three levels of intensity: low (XX = 01), medium (XX = 02) and high (XX = 03)
mg_XX	Network applying *merge* macro operation during its evolution with three levels of intensity: low (XX = 01), medium (XX = 02) and high (XX = 03)
sp_XX	Network applying the *split* macro operation during its evolution with three levels of intensity: low (XX = 01), medium (XX = 02) and high (XX = 03)
sm_eq_XX	Network applying both the *split* and the *merge* macro operations in equal shares during its evolution with three levels of intensity: low (XX = 01), medium (XX = 02) and high (XX = 03)
sm_mm_XX	Network applying both the *split* and the *merge* macro operations during its evolution, with the *merge* one happening more frequently and at three levels of intensity: low (XX = 01), medium (XX = 02) and high (XX = 03)
sm_sm_XX	Network applying both the *split* and the *merge* macro operations during its evolution, with the *split* one happening more frequently and at three levels of intensity: low (XX = 01), medium (XX = 02) and high (XX = 03)

Table 2: Description of the different types of dynamic networks generated for the case study.

where C is the set of communities that an individual encodes; S is each of those communities; ms is the number of edges inside a community S and ns is the number of nodes in community S.

$$C_SCORE = \sum_{S \in C} (\frac{2ms}{ns})^2 \tag{1}$$

The "Average ODF" is an inter-community objective, which calculates how strongly the nodes that belong to different communities are connected. In particular, it measures the average fraction nodes' edges pointing outside their communities. The formula to calculate the Average ODF can be found in Equation 2.Where where C is the set of communities that an individual encodes; S is each of those communities; ns is the number of nodes in the community S and $d(u)$ is the degree of node u.

$$AVG_ODF = \sum_{S \in C} (\frac{1}{ns} \sum_{u \in S} (\frac{|\{u, v | v \notin S\}|}{d(u)})) \tag{2}$$

The MOGA encodes the individuals using the "locus-based adjacency" [54] encoding. In this encoding, each gene is associated to a node of the network and the allele values correspond to nodes of the graph. The genetic operators of the MOGA are the tournament selection, the two-point crossover and a specially-tailored for the problem mutation operation, which randomly selects several genes of an individual and changes their values to a random neighbor community. Finally, the stopping criteria checks the convergence of the algorithm using a Single-Sample Chi-Square Test over the HV of the past 20 Pareto fronts. In addition to this, a maximum number of generations is also used as

OPERATORS	
Encoding	Locus based adjacency
Selection	Tournament
Crossover	Two-point crossover
Mutation	Custom function that randomly selects several genes of an individual and changes their values to a random neighbor community
Stop criteria	Single-Sample Chi-Square Test and max generations

Table 3: MOGA operators

COMMON PARAMETERS	
Tournament size	2
Tournament replace	True
Population size	300
Offspring size	300
Crossover rate	100%
Mutation rate	1% Individual length
Single-Sample Chi-Square	0.05
Max generations	300
RIGA and EIGA	
Number immigrants	10% population

Table 4: MOGA parameters

stopping criteria, in order to assure the end of the execution when the algorithm does not converge. Notice that a summary of the operators used in all MOGAs is available at Table 3

Regarding the hyper-parameters used,which are summarized in Table 4, both the population and offspring sizes are 300 individuals. Please note that the size of the population remains constant over all the evolution process. In addition, the crossover rate is 100% and the mutation rate is set to the 1% of the genes of an individual. Regarding the immigrants, a 10% of the population is replaced with immigrants. Notice that this parameter is only used in the methods were a fixed number of immigrants are introduced into the population. Finally, the alpha used for the Single-Sample Chi-Square Test is 0.05 and the maximum number of generations is 300.

In order to calculate the HV, first the *Community Score* Objective is inverted by subtracting 1.0. Then, the Pareto Fronts are normalized as stipulated in Section 2. Next, the HV is calculated using the ideal reference point $(1.1, 1.1)$. Finally, the number of executions of each MOGA (the M parameter in our methodology) has been set to 20.

3.5 Application of the methodology: analyzing the MOGA in a DCD problem

In this case study, we consider the three immigrant strategies from the state of the art: RIGA, EIGA, and HIGA (which were described in Section 1). In the experimental evaluation, we have also included two additional strategies, in order to set an actual comparison basis:

- Hypermutation Genetic Algorithm (HGA) [55]. This method is not based on the immigrant strategy. This algorithm, instead of continuously inserting immigrants into the population, increases the mutation rate of the population when a change occurs in the environment and then the evolution process continues normally. This method is included in our experimental evaluation since it uses a different strategy than immigrants for dealing with the changes in the environment, which is simpler and less memory consuming since just a single population needs to be stored.
- Delayed Elite Immigrants Genetic Algorithms (DEIGA). This method is based on the immigrant strategy and its similar to EIGA, but with the difference that immigrants are only inserted into the population after a fixed number of generations. This method is included in the study since it allows understanding if the immigrant

Method	Stability	Quality	Speed
RIGA	▼	≈	▲
HIGA	▼	▼	▲
EIGA	▼	▲	▲
HGA	▼	≈	▲
DEIGA	▼	▲	▲

Table 5: Results for the Friedman Signed Ranked Test and the Holm post-hoc using STD as the control method. The first group contains the three immigrant strategies, and the second one the rest of the methods. '▲' indicates that a method performs better than STD. Contrary, '▼' indicates that a method performs worse than STD. Finally, '≈' indicates that the method performs equally to STD.

and native populations explore different sections of the search space. With this method we can analyze if given enough time, both immigrant and native populations converge to the same Pareto fronts or to different ones.

3.5.1 First step: coarse-grain analysis against Standard

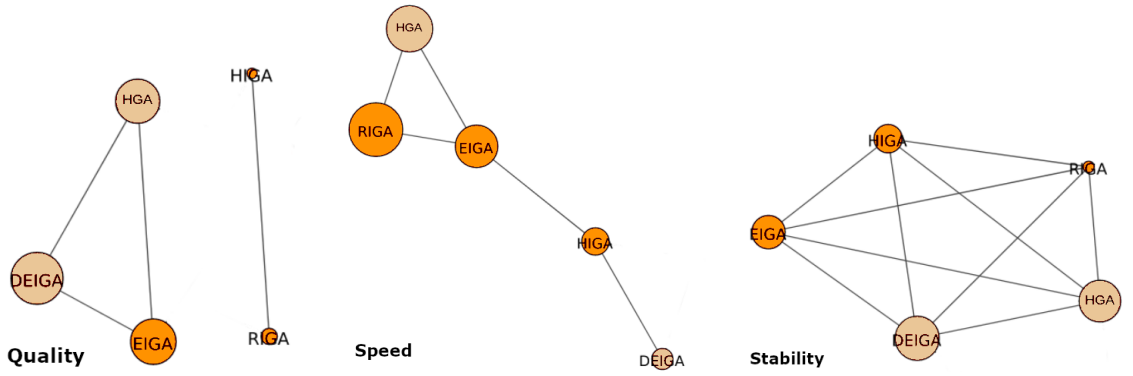
As aforementioned, the first step of the methodology is to compare the performance of the immigrant techniques against the performance of the Standard method. This allows evaluating if the algorithm being tested is able to achieve better results than the base algorithm (without any immigrant schemes).

Table 5 shows the results of the comparative performance of the five methods considered in the study with respect to the *Standard* method. This comparison is based on the usage of the three dimensions proposed in the methodology: quality, speed and stability. The numerical results are mixed with regard to the *quality* of the solutions found. On the one hand, out of the three immigrant techniques under study, EIGA is the only one that improves over *Standard*, while RIGA and HIGA exhibit similar or worse performance, respectively. On the other hand, between the other two methods, DEIGA shows a better performance, while HGA exhibits a similar one. The fact that DEIGA has a better numerical performance than *Standard* was expected as EIGA also does, and DEIGA is just a EIGA with a delay on the insertion of the immigrants. However, the fact that HGA does not improve the results reveals that the gradual insertion of individuals into a population brings benefits to the search mechanism over the direct re-utilization of past populations in DCD problems. Regarding the *speed* of the algorithms, all the techniques studied show a better convergence *speed* than *Standard*. This was an expected result as the main reason for applying a mechanism that reuses past populations is improving the convergence speed by reusing past knowledge. Finally, contrary to what happened with the *speed* dimension, all the algorithms considered in the experimental evaluation present a worse *stability* than the *Standard* algorithm. This implies that the HV of the Pareto Fronts obtained by all the methods have greater variations between consecutive executions compared to the control method. This is a sign that reusing individuals from past populations could deteriorate the exploration capabilities of a MOGA, making it more prone to get stuck in a local minimum/maximum. Please notice that not only do the immigrant strategies suffer from poorer stability but also does the HGA and the Delayed EIGA.

3.5.2 Second step: coarse-grain analysis between immigrant strategies

The second step of the methodology consist in conducting a comparative study between all the techniques under study using the Friedman Signed Ranked Test and Shaffer post-hoc. The study is carried out on the basis of the three dimensions: quality, speed, and stability. The results obtained are presented using graphs in Figure 3. Each node represents a different technique, where the size of the node is inversely proportional to the rank obtained by each method in the Friedman Signed Ranked Test in the Quality dimension and directly proportional in the Stability and Speed ones, i.e., bigger nodes imply a better performance. Two nodes are connected by an edge if the Shaffer post-hoc analysis shows that there is no statistical evidence that the two methods perform differently (i.e., the two methods can be considered equal).

The previous step revealed that EIGA and DEIGA are superior to *Standard* regarding the quality, while RIGA and HGA have a similar performance than the control method. Nonetheless, the second step of the methodology reveals that HGA performs better than RIGA since the HGA node is bigger than the RIGA node in Figure 3 (a) and there is no edge between RIGA and HGA. In addition, this step also allows appreciating that both EIGA and DEIGA have a similar numerical performance (there is an edge between both nodes). From these results, we can conclude that for this problem is better to reuse previous populations with a mutation phase than inserting random immigrants during the evolution



(a) (b) (c)

Figure 3: From left to right, the results of the Friedman Signed Ranked Test and the Shaffer post-hoc analysis for the Quality, Speed, and Stability metrics. Each node represents a different algorithm. The nodes in dark color correspond to the immigrant techniques under study and the light ones to the other methods. The size of the nodes is inversely proportional to its Friedman test rank in the Quality dimension and directly proportional in the Speed and Stability ones (e.i. bigger nodes have a better performance). Two nodes are connected when the difference between two methods are not statistically significant.

process because HGA performs better than RIGA. However, we cannot draw any conclusion about if the immigrant and native population explore different areas of the search space, as both EIGA and DEIGA have a similar performance.

Next, regarding the *Speed* dimension, from the previous step we know that all the methods are faster than *Standard*. However, this step shows that are two different groups (see Figure 3 (b)). The first group is composed of RIGA, HGA, and EIGA, whose nodes form a clique. The second group is composed of HIGA and DEIGA, whose nodes form a kind of tail. From these results, we can conclude that inserting random (RIGA) or elite (EIGA) immigrants exhibit a similar convergence speed than directly re-using previous populations (HGA). Surprisingly, the HIGA method, which uses a combination of elite and random immigrants, presents a worse convergence speed than both of the other immigrant techniques.

Finally, regarding the *Stability* dimension, the previous step of the methodology has shown that all the algorithms considered have worse *stability* than *Standard*. Analyzing the results obtained in this step, shown in Figure 3 (c), we can conclude that all the techniques perform equally bad regarding this dimension being that the the resulting graph forms a clique.

Summarizing, from the results obtained in the first two steps for the problem at stake, it can be concluded that all the immigrant techniques exhibit worse *stability* compared to the *Standard* method and there are no differences between them. In addition, it can also be concluded that gradually inserting elite immigrants into the population has produced better results compared to directly reusing previous populations. However, this is not the case when inserting random immigrants. Surprisingly, the method that combines the use of both elite and random immigrants (HIGA) shows the worse results and even has a worse performance in terms of *quality* than *Standard*. Finally, with the results obtained so far, it can not be evaluated if the native and immigrant populations explore different search space areas, as EIGA and DEIGA have a similar performance in all aspects except for speed (which it was expected due to the delay in the insertion of immigrantes in DEIGA).

3.5.3 Third step: fine-grain analysis

Following the third step of the methodology, the immigrant strategies are evaluated on specific problem instances were difficulties are found. In order to carry out this analysis, first, a study of the quality of the Pareto Fronts found by the algorithms is carried out to find periods where the algorithm is not working properly. Next, an in-depth analysis of those periods is carried out using the *Population mixture visualization technique* (Section 2.3). This visualization technique allows examining what could be causing these difficulties.

Following the methodology, first, the performance of the immigrant techniques for each dynamic network generated is evaluated on the basis of the Quality metric to identify which immigrant strategies are performing poorly. Table 6 shows the quality obtained by each immigrant method compared to the standard method for each problem instance tested. A green triangle pointing up (▲) indicates that a method performs better than STD. Contrary, a red triangle

Instance	HIGA	RIGA	EIGA	HGA	DEIGA
mg_01	▼	▼	≈	≈	≈
mg_02	▼	▼	≈	≈	≈
mg_03	▼	▼	≈	≈	≈
mx_01	≈	≈	▲	▲	▲
mx_02	≈	≈	▲	▲	▲
mx_03	≈	≈	▲	▲	▲
sm_eq_01	▼	▼	▼	▼	≈
sm_eq_02	▼	▼	▼	▼	≈
sm_eq_03	▼	≈	▲	▲	▲
sm_mm_01	▼	≈	▲	≈	≈
sm_mm_02	▼	▼	▼	▼	≈
sm_mm_03	≈	≈	▲	▲	▲
sm_sm_01	▼	▼	≈	≈	▲
sm_sm_02	▼	▼	≈	≈	≈
sm_sm_03	▼	≈	▲	≈	▲
sp_01	▼	≈	▲	▲	▲
sp_02	▼	≈	▲	≈	▲
sp_03	▼	▼	≈	≈	≈

Table 6: Results for the Friedman Signed Ranked Test and the Holm post-hoc on the HV using STD as the control method.

pointing down (▼) indicates that a method performs worse than STD. Finally, the approximated sign (≈) indicates that the method performs equally to STD. The evaluation has been done using the Friedman Signed Ranked Test and the Holm post-hoc analysis using as data the 20 Quality values (one per execution of a problem instance) obtained by each strategy. Each quality value is calculated by doing the median of the HV obtained at each of the nine snapshots that form a dynamic network.

The results from Table 6 reinforce the conclusions drawn in the two previous steps, using elite immigrants does have benefits over reusing past populations, while using random immigrants does not. It should be noted that HIGA and RIGA have a worse numerical performance than HGA, which in turn performs worse than EIGA. However, it can be seen that there are several networks for which all of the immigrant techniques under study obtain worse *quality* than the baseline. In particular, the networks *sm_eq_01*, *sm_eq_02*, and *sm_mm_01* are of special interest because all the methods fail except DEIGA. To further analyze the previous phenomena, the rest of the section studies the behavior of each of the immigrant techniques on the *sm_eq_01* network. Only the results of the *sm_eq_01* network are shown because the conclusion obtained by analyzing the others are similar.

With the above in mind, in a second step, the attainment surfaces and the HV evolution of the Pareto Fronts found at each snapshot of the network *sm_eq_01* are evaluated. This allows identifying in which snapshots the algorithms are working properly in which they don't. The network under study is composed of nine snapshots. All the methods included in the study perform well in all the snapshots, except for the last three. We focus our analysis on the results of the seventh snapshot, even though the same effect could be found in the other two ones.

As aforementioned, Figure 4 only shows the attainment surfaces and the HV evolution of the immigrant techniques obtained by all the immigrant techniques in the seventh snapshot of the *sm_eq_01* network. The attainment surfaces plot shows that all the techniques except for DEIGA perform worse than *Standard*; notice the dark areas on the left attainment surfaces. Moreover, in the HV evolution plots, it can be seen that all immigrant techniques except for DEIGA suffer from premature convergence to a greater or lesser extent. It should be noted that the orange line of the HV plots gets stuck in a local minimum compared to the blue one. To further study why the different immigrant techniques are not preventing this premature convergence, each immigrant technique is studied independently using the populations mixture visualization technique:

- Starting with the RIGA technique, the Figure 5 shows the number of *half-blood* (n+i) individuals (i.e., an offspring of a native and an immigrant individual) that continues from one generation to the next (in orange) and that are removed from the population (in blue). The different sub-figures in Figure 5 correspond to different percentages of immigrants inserted into the population at each generation; 1% (3 individuals), 5% (15 individuals), 10% (30 individuals) and 20% (60 individuals) of the population are replaced by immigrants is shown in sub-figures (a), (b), (c) , and (d), respectively. As it is shown in the figure, the mixture between natives

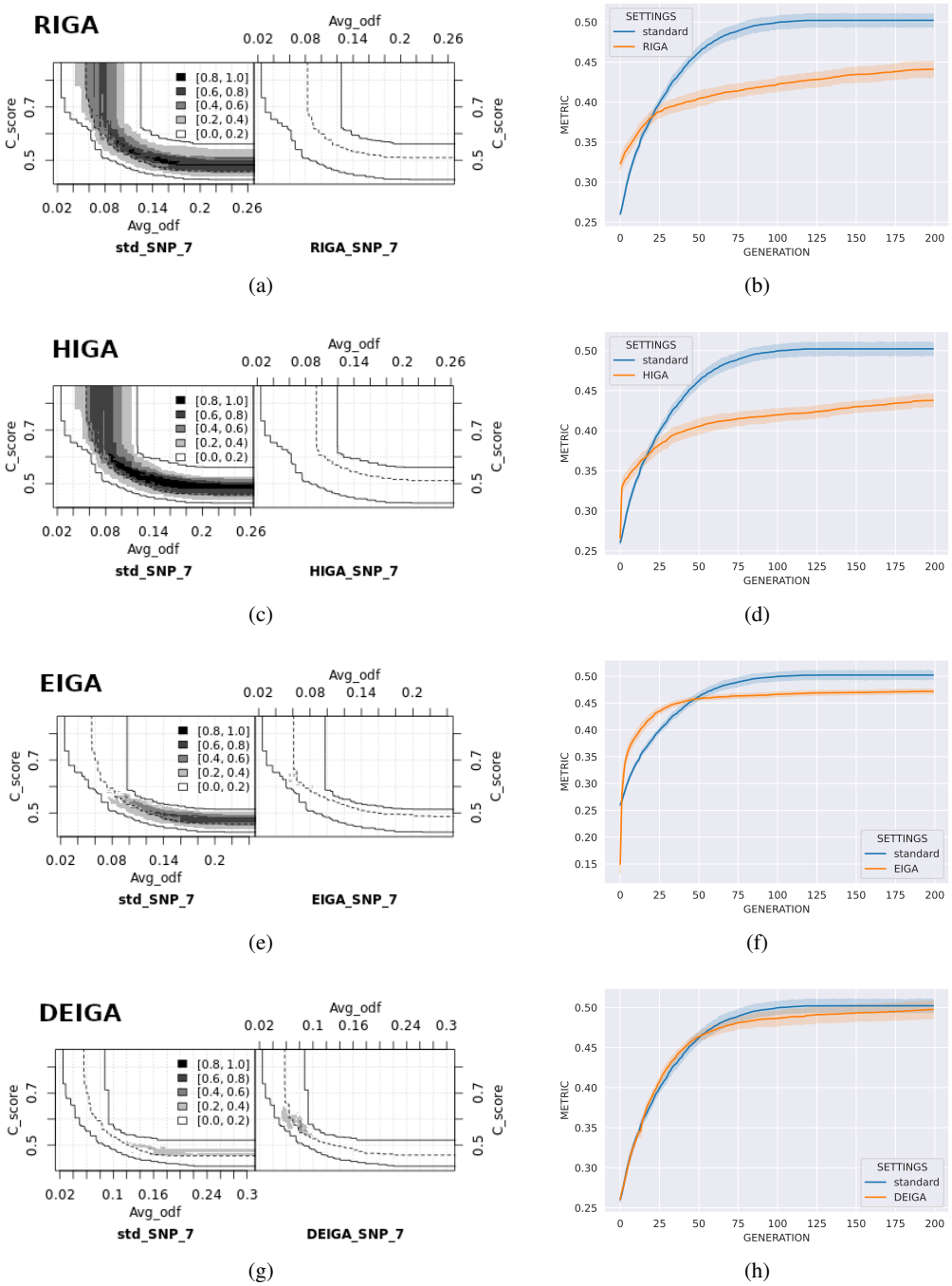


Figure 4: Left figures (a, c, e, g) shows the attainment surfaces of the final Pareto Fronts found by the STD method and the immigrant strategies. The STD method is shown on the left panel of the attainment surfaces. Right figures (b, d, f, h) shows the evolution of the HV over the different generations of the MOGA for the same methods. The STD method is shown in orange, while the immigrant strategy is shown in blue.

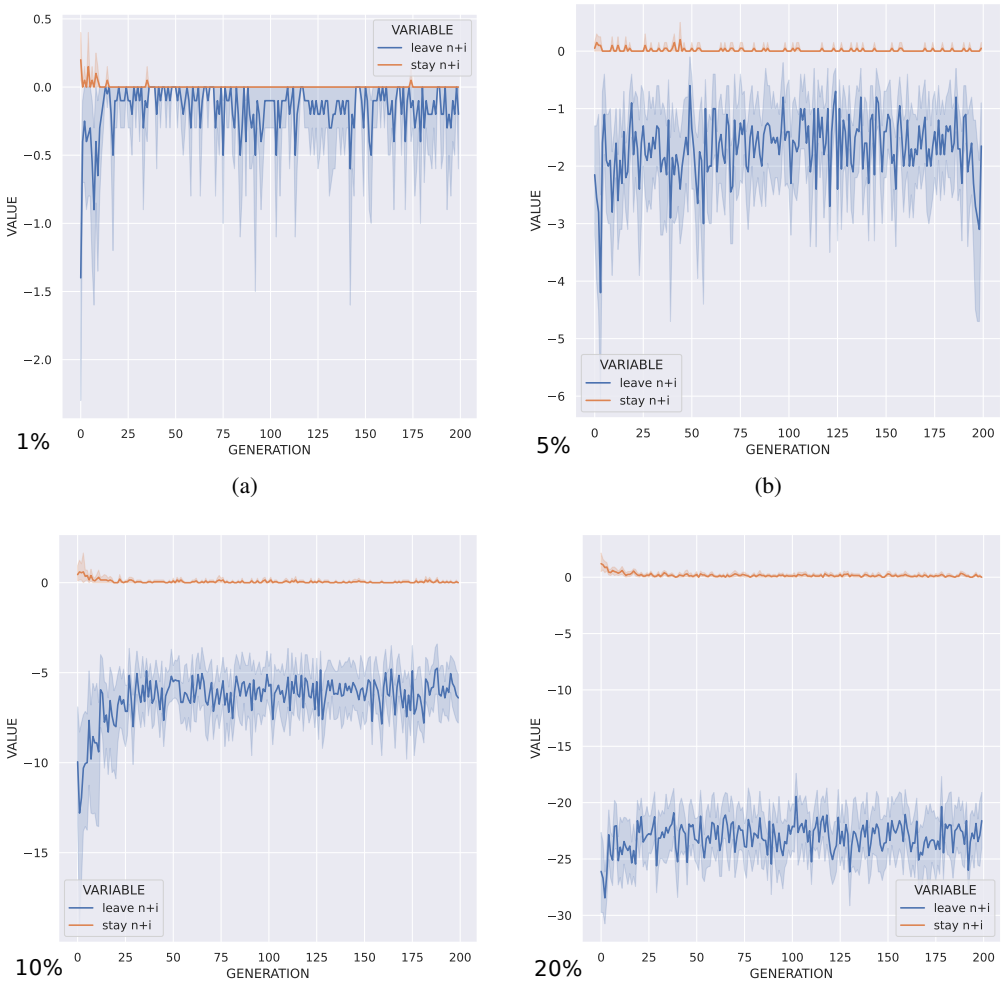
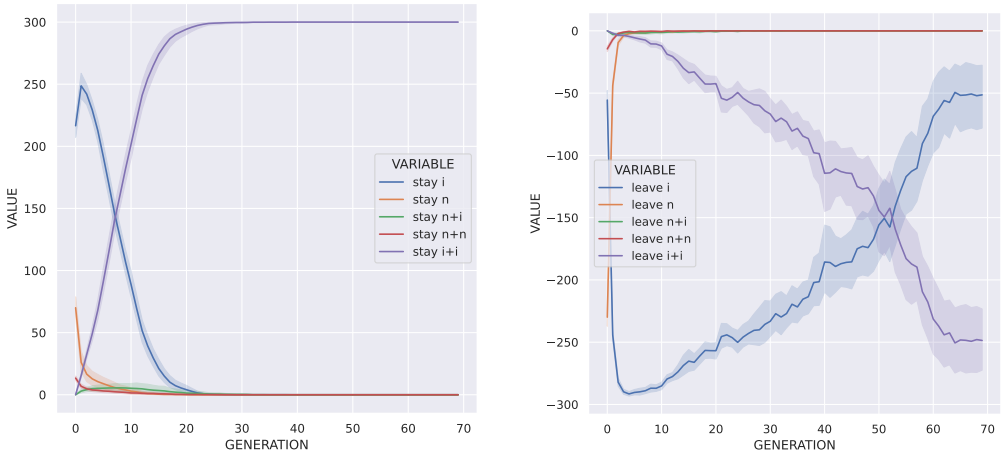


Figure 5: Number of *half-blood* individuals for the RIGA method that survive (in orange) and are expelled (in blue) of the population between two generations. The X axis shows the number of generation and the Y axis the number of individuals of a particular type. The percentage on the bottom right corner of the plot indicates the percentage of immigrants inserted into the population at each generation.

(individuals from the previous snapshot in the case of RIGA) and immigrants (completely random individuals in the case of RIGA) have not enough quality and are almost completely removed from the population in the same generation that are created. Moreover, the percentage of immigrants inserted into the population does not change this effect. Therefore, it is hard that two immigrant individuals mate in RIGA and almost impossible that their offsprings continue to the next generation. Since the immigrants are just being expelled from the population without having a chance of passing their genes to the next generation, they are essentially wasting space on the population. As a result, in practice, the RIGA technique in the DCD problem is equivalent to having a smaller population, which produces the fast convergence speed and the low quality of the solutions already found in Section 3.5.1.

- Regarding the HIGA technique, Figure 6 shows the number of individuals of the different types during the evolutionary process for the HIGA method. The figure on the left (sub-figure (a)) shows the number of individuals that continue from one generation to the next, while the one on the right (sub-figure (b)) shows the ones that are being expelled from the population. The sub-figure (a) shows that the HIGA method inserts a large number of immigrants (individuals from the previous snapshot) during the first generations of the evolution. In fact, it inserts so many immigrants that the native population becomes extinct after the fifth



(a) (b)

Figure 6: Number of individuals of different types for the HIGA method. The X axis show the number of generation during the evolutionary process and the Y axis the number of individuals of a particular type. Sub-figure (a) shows the number of individuals that survive from one generation to the next. Sub-figure (b) shows the number of individuals that are expelled from the population.

generation. Nevertheless, analyzing the sub-figure (b), it can be noticed that after the native population has become extinct, the number of immigrant individuals that leave the population decreases and then plateaus. It is not unreasonable to hypothesize that this plateau is produced by the algorithm trying to enhance its exploration capabilities by adding random immigrants, which is the same technique that RIGA uses. Summarizing, the HIGA method is too aggressive at the beginning of the evolutionary process and removes all the native individuals from the population, essentially behaving like HGA. After that, it tries to enhance the population diversity by introducing random individuals, just like RIGA, which results in the immigrants just using space and not contributing to the exploration (i.e., it is like using a smaller population). This behavior explains the bad quality of the results of HIGA, just like RIGA. However, contrary to RIGA, HIGA does it gradually during the evolutionary process, hence the lower convergence speed.

- With respect to EIGA, Figure 7 shows the number of individuals of different types for a particular generation. Each sub-figure shows how the different types vary for distinct percentages of immigrant individuals. Contrary to the other methods, in EIGA there is an adequate mixture between native and immigrant individuals, which can be noticed in the half-blood line (the n+i green line). As expected, there is a transition phase where the two populations (immigrant and native) mate, producing valid offsprings. After that period, both populations continue mixing, but they result in solutions with lower quality that are expelled from the population. The length of this mixing period varies depending on the percentage of immigrant individuals inserted into the population, the higher the percentage of immigrants, the shorter the period. Furthermore, for percentages as high as 5%, the immigrant population (or their descendants) take over the population (sub-figures (b), (c), and (d)). However, when the percentage is only 1% (subfigure (a)), it can be seen that no population fully dominates the other. The evolutionary process reaches an equilibrium where the immigrant population (or their descendants) occupies a part of the population, while the native one occupies the rest. This fact suggest that the immigrant and native population could explore different regions of the landscape. Summarizing, the EIGA strategy is the only immigrant technique from the study that achieves an adequate mixture between populations, hence its better results and fast convergence speed, combining the quality of HGA with the speed of convergence of RIGA.
- Finally, we analyze DEIGA, which was included for comparative reasons. As aforementioned, this method works like EIGA, but the insertion of immigrants is delayed a certain number of generations. From the results shown in Figure 7 sub-figure (a), it can be hypothesized that if the native population is allowed to evolve for a certain number of generations, so they can better adapt to the environment, the mixture of individuals could produce better results. DEIGA method allows checking this hypothesis. Figure 8 shows the number of individuals of different types during the evolutionary process for DEIGA. Each sub-figure shows how the different types varies when waiting for a different number of generations, which is shown at the top right

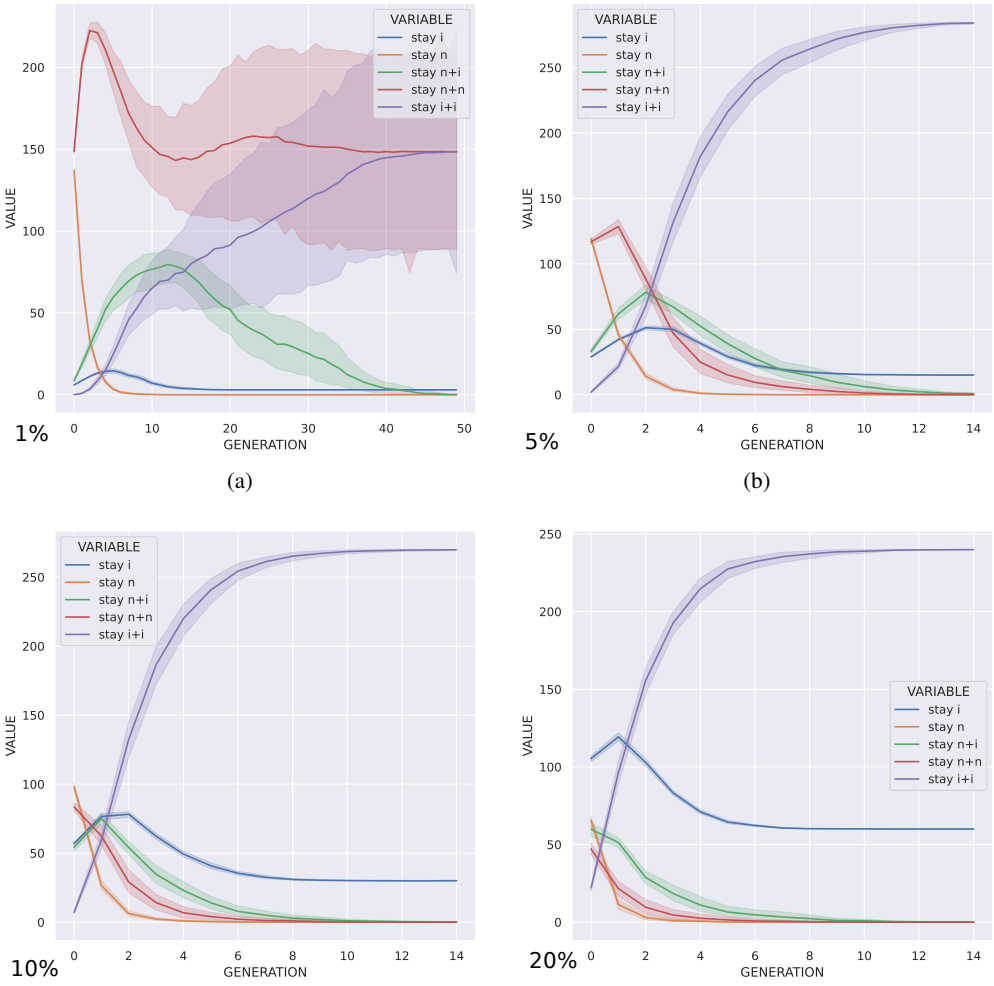


Figure 7: Number of individuals of different types for the EIGA method. The X axis show the number of generation during the evolutionary process and the Y axis the number of individuals of a particular type. The percentage on the top right corner of the plot indicates the percentage of immigrants inserted into the population at each generation.

corner. In all plots a 10% of immigrant individuals are inserted on each generation. As it can be seen, even waiting a small number of generations (e.g., 10) prevents the immigrant population from taking over the whole population (notice the difference with Figure 7 (c)). Also, the larger the number of generations DEIGA waits to insert the immigrants, the smaller is the size difference between the populations when an equilibrium is reached. However, if the value is too large, as in the sub-figure (e), the opposite effect is produced since the immigrants of the population have not enough quality to occupy their own space in the population. Nevertheless, when the immigrant solutions enter the population, there is always a mixture between both type of individuals (the n+i green line is present in all the plots). Summarizing, letting the population evolve some generations before inserting immigrant solutions produces the best results as prevents the immigrants from taking over all the population and allows a good mixture between populations. However, waiting too many generations produces the opposite effect and prevents the immigrant solutions from entering into the population. Nevertheless, even in those cases, some kind of mixture between the populations always appears.

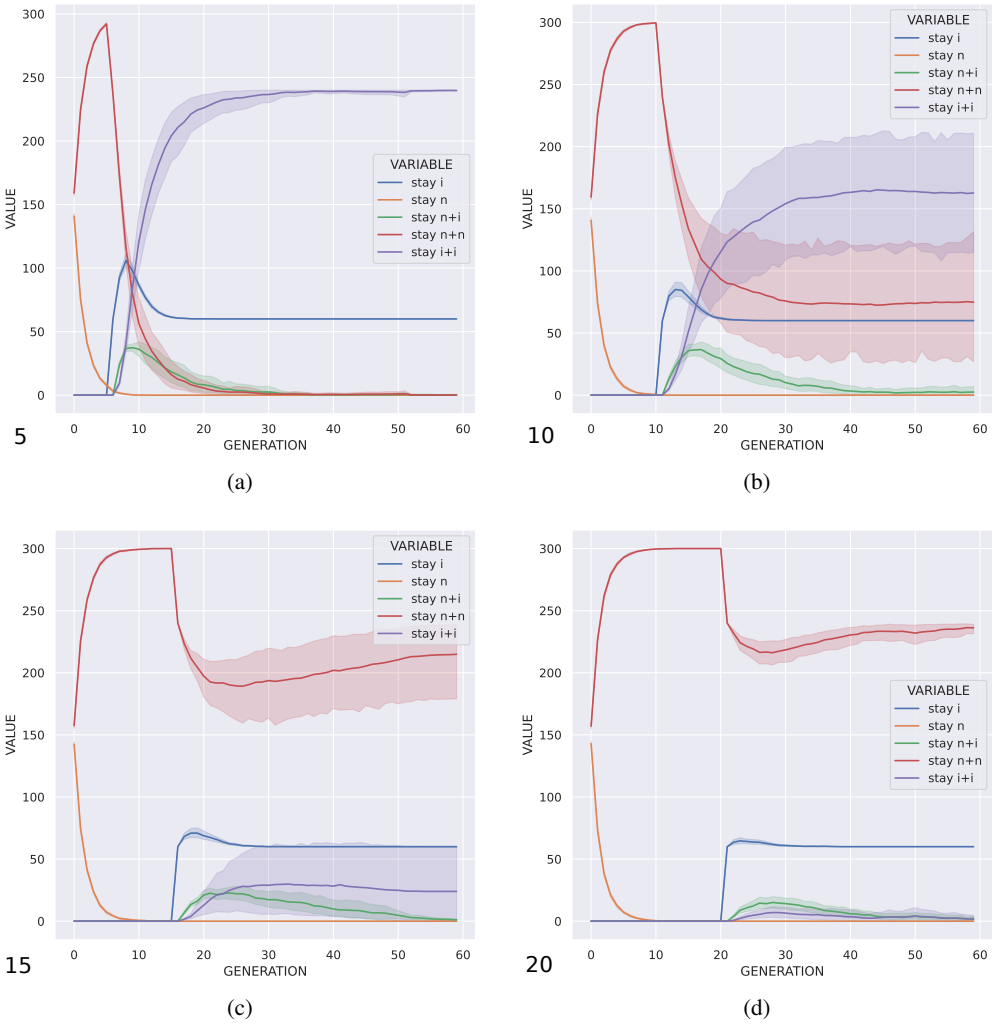


Figure 8: Number of individuals of different types for the DEIGA method. The X axis shows the number of generation during the evolutionary process whereas the Y axis shows the number of individuals of a particular type. The number on the top right corner of the plot indicates the number of generations waited until the immigrants are inserted into the population. In all plots a 10% of immigrant individuals are inserted at each generation.

3.6 Conclusions of the experimental analysis

The experimental evaluation has shown that both RIGA and HIGA have a low numerical performance. The fine-grain analysis carried out revealed that the insertion of completely random individuals into an already established population produces offspring of so little quality, compare to the rest, that they are immediately discarded from the population. In practice, these methods are reducing the size of the population since the immigrant solutions are discarded without contributing to the exploratory capabilities of the algorithm. However, the EIGA method shows better results as it is the only method where an adequate mixture between the native and immigrant populations is achieved. Nevertheless, the percentage of immigrants inserted in the population affects the quality of the numerical results. If too many immigrants are inserted, the selection pressure of them is too high and they over take the whole population, making the native solutions extinct. Surprisingly, when the percentage of immigrants inserted is not too high, a equilibrium is achieved and both population continue evolving independently.

To further study this phenomenon the EIGA algorithm was evaluated including a delay in the insertion of immigrants (DEIGA). These experiments have shown that waiting as little as ten generations before inserting the immigrants produces an equilibrium where the immigrants and the native population continue evolving independently, even for high percentages of immigrant insertions like 10% of the population and the further the algorithm waits, the lower is the different in size between both populations during the equilibrium. However, if the algorithm waits too many generations, the opposite effect happens and the immigrant individuals have not enough quality to enter the population. Nevertheless, a mixture between populations is always present.

The appearance of this equilibrium where both populations evolve in parallel, suggest that, for this particular setup (multi-objective genetic algorithms for dynamic community detection), the immigrant and the native solutions have different exploration capabilities and in order to achieve the best results the population needs some time to mature. For this reason, it is crucial that the selection pressure of the algorithm does not expel the native solutions from the population too fast, before the individuals are exposed to other type of individuals.

However, the results presented in this section should be considered preliminary for the DCD field, as they have only been tested using synthetic networks. For more robust results, benchmarks composed of real networks are also needed. However, at the time of writing, finding them is not an easy task. In addition, all the conclusions presented here are only relevant when estimating the capability of a method to find non-dominated sets of solutions. Which solutions should be selected from each non-dominated set to form a good dynamic community is still an open question in the area of dynamic community detection. Depending on the criteria followed, the results could change. Hence, we cannot assure that the method that generates better non-dominated fronts will give the best community dynamics. At least until some good criteria is found and tested. However, as there is no defined criteria jet, is not a far stretch if we hypothesize that having the richest and most varied set of solutions will be a good start for any future criteria.

4 Conclusions and future work

This article presents a new methodology for evaluating immigrant strategies in MOGAs on the basis of three different dimensions: Quality, Stability, and Speed. The purpose of this methodology is twofold. On the one hand, it allows evaluating if a particular immigrant strategy is improving o hindering one of the aforementioned dimensions. On the other hand, the methodology also allows obtaining insights about what could be happening in the case that some dimensions get hindered.

To prove the effectiveness of the new methodology, a case study has been carried out on the Dynamic Community Detection (DCD) problem. In our case study, we analyze the results of the three most common immigrant strategies: RIGA, EIGA, and HIGA. In addition, two other methods were included, HGA and DEIGA, in order to set an actual comparison basis.

The experimental analysis has shown that inserting random immigrants into an already established population produces low quality offspring that do not survive. In practice, for this particular case study, using random immigrants is equivalent to reducing the population size and it is better to directly re-use a previous population. As a consequence, RIGA and HIGA have a low numerical performance, even when compared with HGA. On the contrary, the study shows that using elite immigrants achieves promising results and produces benefits over the direct re-utilization of previous populations. Hence, the better performance of EIGA over all other methods, including the comparison basis method HGA. In addition, the experimental analysis has shown that EIGA reaches an equilibrium between both immigrant and native solutions in the population, suggesting that a multi-population genetic algorithm would be beneficial for this particular strategy on this particular problem. However, it is important to note that the results extracted from the case study should be considered preliminary, as they were only tested using synthetic networks. More robust results require benchmarks composed of real networks, which, at the time of writing, are difficult to find. However, despite this

limitation, the results of the case study should be considered robust when assessing the reliability of the methodology, which is the main objective of the article. Therefore, after the above analysis, we can conclude that the proposed methodology has allowed systematizing the analysis of the different immigrant techniques and that the population mixture visualization technique provides valuable insight into their inner workings. Finally, with the above in mind, we can declare that our methodology could ease the successful application of immigrant techniques to MOGAs.

However, three areas that deserve further study are identified in this work. From the dynamics of the interactions between native and immigrant solutions found in EIGA, a first issue is to design a multi-population multi-objective genetic algorithm for the Dynamic Community Detection problem. This algorithm would have two populations, one with the native solutions and one with the immigrants, that work independently but periodically exchange solutions between the populations. Alternatively, it would also be interesting to study the DCD problem from the perspective of an Evolutionary Multi-Task Optimisation (EMTO). In EMTO, instead of solving each task one at a time, all tasks are solved at the same time using the knowledge gained in one task to help the others. In particular, is of special interest the MFEA [56] algorithm in which all tasks (snapshots in DCD) are optimized at the same time by a single population. Secondly, and since the DCD problem is a discrete problem, we aim to extend this type of analysis to more complex dynamic problems like continuous problems or tackling other dynamic multi-objective problems to completely asses the evidence of the benefits of the analysis methodology proposed in this article. Finally, since the proposed methodology allows systematizing the analysis in the case of dynamic multi-objective problems, we found interesting to adapt it first to dynamic mono-objective problems, and later to other related problems like EMTO.

Acknowledgements

This work has been funded by Grant PLEC2021-007681 (XAI-DisInfodemics) and PID2020-117263GB-100 (FightDIS) funded by MCIN/AEI/ 10.13039/501100011033 and, as appropriate, by "ERDF A way of making Europe", by the "European Union" or by the "European Union NextGenerationEU/PRTR", by the research project CIVIC: Intelligent characterisation of the veracity of the information related to COVID-19, granted by BBVA FOUNDATION GRANTS FOR SCIENTIFIC RESEARCH TEAMS SARS-CoV-2 and COVID-19, by European Comission under IBERIFIER - Iberian Digital Media Research and Fact-Checking Hub (2020-EU-IA-0252). Martín Pedemonte research stay at Universidad Politécnica de Madrid was funded by grants from ANII - MOV_CA_2019_1_156657 and CSIC, UDELAR. Finally, David Camacho has been supported by the Comunidad Autónoma de Madrid under: "Convenio Plurianual with the Universidad Politécnica de Madrid in the actuation line of *Programa de Excelencia para el Profesorado Universitario*".

References

[1] Ganesan Ramalingam and Thomas Reps. On the computational complexity of dynamic graph problems. *Theoretical Computer Science*, 158(1-2):233–277, 1996.

[2] Jürgen Branke and Hartmut Schmeck. Designing evolutionary algorithms for dynamic optimization problems. In *Advances in evolutionary computing*, pages 239–262. Springer, 2003.

[3] Marco Farina, Kalyanmoy Deb, and Paolo Amato. Dynamic multiobjective optimization problems: test cases, approximations, and applications. *IEEE Transactions on evolutionary computation*, 8(5):425–442, 2004.

[4] Xin Yu, Yaochu Jin, Ke Tang, and Xin Yao. Robust optimization over time—a new perspective on dynamic optimization problems. In *IEEE Congress on evolutionary computation*, pages 1–6. IEEE, 2010.

[5] Shengxiang Yang. Evolutionary computation for dynamic optimization problems. In *Proceedings of the 15th annual conference companion on Genetic and evolutionary computation*, pages 667–682, 2013.

[6] Trung Thanh Nguyen, Shengxiang Yang, and Juergen Branke. Evolutionary dynamic optimization: A survey of the state of the art. *Swarm and Evolutionary Computation*, 6:1–24, 2012.

[7] Kay Chen Tan, Liang Feng, and Min Jiang. Evolutionary transfer optimization-a new frontier in evolutionary computation research. *IEEE Computational Intelligence Magazine*, 16(1):22–33, 2021.

[8] Tingyang Wei, Shibin Wang, Jinghui Zhong, Dong Liu, and Jun Zhang. A review on evolutionary multi-task optimization: Trends and challenges. *IEEE Transactions on Evolutionary Computation*, 2021.

[9] Radhia Azzouz, Slim Bechikh, and Lamjed Ben Said. Dynamic multi-objective optimization using evolutionary algorithms: a survey. In *Recent advances in evolutionary multi-objective optimization*, pages 31–70. Springer, 2017.

[10] Helen G Cobb and John J Grefenstette. Genetic algorithms for tracking changing environments. Technical report, Naval Research Lab Washington DC, 1993.

[11] Anabela Simões and Ernesto Costa. An immune system-based genetic algorithm to deal with dynamic environments: diversity and memory. In *Artificial Neural Nets and Genetic Algorithms*, pages 168–174. Springer, 2003.

[12] Keehyung Kim, Robert Ian McKay, and Byung-Ro Moon. Multiobjective evolutionary algorithms for dynamic social network clustering. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, pages 1179–1186. ACM, 2010.

[13] Shengxiang Yang. Memory-based immigrants for genetic algorithms in dynamic environments. In *Proceedings of the 7th annual conference on Genetic and evolutionary computation*, pages 1115–1122, 2005.

[14] Shengxiang Yang. Genetic algorithms with memory-and elitism-based immigrants in dynamic environments. *Evolutionary Computation*, 16(3):385–416, 2008.

[15] Hui Cheng and Shengxiang Yang. Genetic algorithms with immigrants schemes for dynamic multicast problems in mobile ad hoc networks. *Engineering Applications of Artificial Intelligence*, 23(5):806–819, 2010.

[16] Xin Yu, Ke Tang, Tianshi Chen, and Xin Yao. Empirical analysis of evolutionary algorithms with immigrants schemes for dynamic optimization. *Memetic Computing*, 1(1):3–24, 2009.

[17] Renato Tinós and Shengxiang Yang. A self-organizing random immigrants genetic algorithm for dynamic optimization problems. *Genetic Programming and Evolvable Machines*, 8(3):255–286, 2007.

[18] Shengxiang Yang. Genetic algorithms with elitism-based immigrants for changing optimization problems. In *Workshops on Applications of Evolutionary Computation*, pages 627–636. Springer, 2007.

[19] Shengxiang Yang and Renato Tinós. A hybrid immigrants scheme for genetic algorithms in dynamic environments. *International Journal of Automation and Computing*, 4(3):243–254, 2007.

[20] Enrique Alba and Bernabé Dorronsoro. *Cellular genetic algorithms*, volume 42. Springer, 2009.

[21] Mario Giacobini, Marco Tomassini, Andrea GB Tettamanzi, and Enrique Alba. Selection intensity in cellular evolutionary algorithms for regular lattices. *IEEE Transactions on Evolutionary Computation*, 9(5):489–505, 2005.

[22] Martin Pedemonte, Francisco Luna, and Enrique Alba. A theoretical and empirical study of the trajectories of solutions on the grid of systolic genetic search. *Information Sciences*, 445:97–117, 2018.

[23] Ronald W Morrison. Performance measurement in dynamic environments. In *GECCO workshop on evolutionary algorithms for dynamic optimization problems*, number 5-8. Citeseer, 2003.

[24] Krzysztof Trojanowski and Zbigniew Michalewicz. Evolutionary algorithms for non-stationary environments. In *Proc. of 8th workshop: Intelligent information systems*, pages 229–240. Citeseer, 1999.

[25] Karsten Weicker. Performance measures for dynamic environments. In *International Conference on Parallel Problem Solving from Nature*, pages 64–73. Springer, 2002.

[26] Michalis Mavrovouniotis and Shengxiang Yang. Genetic algorithms with adaptive immigrants for dynamic environments. In *2013 IEEE Congress on Evolutionary Computation*, pages 2130–2137. IEEE, 2013.

[27] Eckart Zitzler and Lothar Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE transactions on Evolutionary Computation*, 3(4):257–271, 1999.

[28] Joaquín Derrac, Salvador García, Daniel Molina, and Francisco Herrera. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation*, 1(1):3–18, 2011.

[29] Manuel López-Ibánez, Luís Paquete, and Thomas Stützle. Exploratory analysis of stochastic local search algorithms in biobjective optimization. In *Experimental methods for the analysis of optimization algorithms*, pages 209–222. Springer, 2010.

[30] Angel Panizo-LLedot, Gema Bello-Orgaz, and David Camacho. A multi-objective genetic algorithm for detecting dynamic communities using a local search driven immigrant's scheme. *Future Generation Computer Systems*, 110:960–975, 2020.

[31] Aaron Clauset, Mark EJ Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical review E*, 70(6):066111, 2004.

[32] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008, 2008.

[33] Vincent A Traag, Ludo Waltman, and Nees Jan Van Eck. From louvain to leiden: guaranteeing well-connected communities. *Scientific reports*, 9(1):1–12, 2019.

[34] Michele Coscia, Giulio Rossetti, Fosca Giannotti, and Dino Pedreschi. Demon: a local-first discovery method for overlapping communities. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 615–623, 2012.

[35] Eneko Osaba, Javier Del Ser, David Camacho, Miren Nekane Bilbao, and Xin-She Yang. Community detection in networks using bio-inspired optimization: Latest developments, new results and perspectives with a selection of recent meta-heuristics. *Applied Soft Computing*, 87:106010, 2020.

[36] Gema Bello-Orgaz and David Camacho. Evolutionary clustering algorithm for community detection using graph-based information. In *2014 IEEE congress on evolutionary computation (CEC)*, pages 930–937. IEEE, 2014.

[37] Gema Bello-Orgaz, Sancho Salcedo-Sanz, and David Camacho. A multi-objective genetic algorithm for overlapping community detection based on edge encoding. *Information Sciences*, 462:290–314, 2018.

[38] Martín Pedemonte, Ángel Panizo-LLedot, Gema Bello-Orgaz, and David Camacho. Exploring multi-objective cellular genetic algorithms in community detection problems. In *International Conference on Intelligent Data Engineering and Automated Learning*, pages 223–235. Springer, 2020.

[39] David Camacho, Ángel Panizo-LLedot, Gema Bello-Orgaz, Antonio Gonzalez-Pardo, and Erik Cambria. The four dimensions of social network analysis: An overview of research methods, applications, and software tools. *Information Fusion*, 63:88–120, 2020.

[40] Deepayan Chakrabarti, Ravi Kumar, and Andrew Tomkins. Evolutionary clustering. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 554–560. ACM, 2006.

[41] Francesco Folino and Clara Pizzuti. An evolutionary multiobjective approach for community discovery in dynamic networks. *IEEE Transactions on Knowledge and Data Engineering*, 26(8):1838–1852, 2013.

[42] Guoqiang Chen, Yuping Wang, and Jingxuan Wei. A new multiobjective evolutionary algorithm for community detection in dynamic complex networks. *Mathematical problems in engineering*, 2013, 2013.

[43] A Attea Bara'a and Haidar S Khoder. A new multi-objective evolutionary framework for community mining in dynamic social networks. *Swarm and Evolutionary Computation*, 31:90–109, 2016.

[44] Jingjing Ma, Jie Liu, Wenping Ma, Maoguo Gong, and Licheng Jiao. Decomposition-based multiobjective evolutionary algorithm for community detection in dynamic social networks. *The Scientific World Journal*, 2014, 2014.

[45] Kalyanmoy Deb, S Karthik, et al. Dynamic multi-objective optimization and decision-making using modified nsga-ii: A case study on hydro-thermal power scheduling. In *International conference on evolutionary multi-criterion optimization*, pages 803–817. Springer, 2007.

[46] Qingfu Zhang and Hui Li. Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on evolutionary computation*, 11(6):712–731, 2007.

[47] Ying Yin, Yuhai Zhao, He Li, and Xiangjun Dong. Multi-objective evolutionary clustering for large-scale dynamic community detection. *Information Sciences*, 549:269–287, 2021.

[48] Xu Zhou, Xiaohui Zhao, and Yanheng Liu. A multiobjective discrete bat algorithm for community detection in dynamic networks. *Applied Intelligence*, 48(9):3081–3093, 2018.

[49] Chuan Shi, Philip S Yu, Zhenyu Yan, Yue Huang, and Bai Wang. Comparison and selection of objective functions in multiobjective community detection. *Computational Intelligence*, 30(3):562–582, 2014.

[50] Oualid Benyahia, Christine Largeron, Baptiste Jeudy, and Osmar R Zaïane. Dancer: Dynamic attributed network with community structure generator. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 41–44. Springer, 2016.

[51] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002.

[52] Clara Pizzuti. Ga-net: A genetic algorithm for community detection in social networks. In *International Conference on Parallel Problem Solving from Nature*, pages 1081–1090. Springer, 2008.

[53] Gary William Flake, Steve Lawrence, and C Lee Giles. Efficient identification of web communities. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 150–160. ACM, 2000.

[54] YoungJa Park and ManSuk Song. A genetic algorithm for clustering problems. In *Proceedings of the third annual conference on genetic programming*, volume 1998, pages 568–575, 1998.

[55] Johnny Kelsey and Jon Timmis. Immune inspired somatic contiguous hypermutation for function optimisation. In *Genetic and Evolutionary Computation Conference*, pages 207–218. Springer, 2003.

[56] Abhishek Gupta, Yew-Soon Ong, and Liang Feng. Multifactorial evolution: toward evolutionary multitasking. *IEEE Transactions on Evolutionary Computation*, 20(3):343–357, 2015.

Bibliografía

[1] H. Alvari, A. Hajibagheri, G. Sukthankar, and K. Lakkaraju. Identifying community structures in dynamic networks. *Social Network Analysis and Mining*, 6(1):77, 2016.

[2] D. R. Amancio, O. N. Oliveira Jr, and L. da F Costa. Robustness of community structure to node removal. *Journal of Statistical Mechanics: Theory and Experiment*, 2015(3):P03003, 2015.

[3] S. Asur, S. Parthasarathy, and D. Ucar. An event-based framework for characterizing the evolutionary behavior of interaction graphs. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 3(4):16, 2009.

[4] T. Aynaud and J.-L. Guillaume. Multi-step community detection and hierarchical time segmentation in evolving networks. In *Proceedings of the 5th SNA-KDD workshop*, 2011.

[5] J. Bader and E. Zitzler. Hype: An algorithm for fast hypervolume-based many-objective optimization. *Evolutionary computation*, 19(1):45–76, 2011.

[6] S. Banerjee and B. Pal. On the enumeration of maximal (δ, γ)-cliques of a temporal network. In *Proceedings of the ACM India Joint International Conference on Data Science and Management of Data*, pages 112–120. ACM, 2019.

[7] A. A. Bara'a and H. S. Khoder. A new multi-objective evolutionary framework for community mining in dynamic social networks. *Swarm and Evolutionary Computation*, 31:90–109, 2016.

[8] C. N. Bendtsen and T. Krink. Dynamic memory model for non-stationary optimization. In *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No. 02TH8600)*, volume 1, pages 145–150. IEEE, 2002.

[9] S. Y. Bhat and M. Abulaish. Octracker: A density-based framework for tracking the evolution of overlapping communities in osns. In *Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2012)*, pages 501–505. IEEE Computer Society, 2012.

[10] S. Boudebza, R. Cazabet, F. Azouaou, and O. Nouali. Olcpm: An online framework for detecting overlapping communities in dynamic social networks. *Computer Communications*, 123:36–51, 2018.

[11] J. Branke, T. Kaußler, C. Smidt, and H. Schmeck. A multi-population approach to dynamic optimization problems. In *Evolutionary design and manufacture*, pages 299–307. Springer, 2000.

[12] P. Bródka, S. Saganowski, and P. Kazienko. Ged: the method for group evolution discovery in social networks. *Social Network Analysis and Mining*, 3(1):1–14, 2013.

[13] D. Camacho, A. Panizo-LLedot, G. Bello-Orgaz, A. Gonzalez-Pardo, and E. Cambria. The four dimensions of social network analysis: An overview of research methods, applications, and software tools. *Information Fusion*, 63:88–120, 2020.

[14] R. Cazabet and F. Amblard. *Dynamic Community Detection*, pages 404–414. Springer New York, New York, NY, 2014.

[15] R. Cazabet, F. Amblard, and C. Hanachi. Detection of overlapping communities in dynamical social networks. In *2010 IEEE second international conference on social computing*, pages 309–314. IEEE, 2010.

[16] R. Cazabet, S. Boudebza, and G. Rossetti. Evaluating community detection algorithms for progressively evolving graphs. *Journal of Complex Networks*, 8(6):cnaa027, 2020.

[17] R. Cazabet, G. Rossetti, and F. Amblard. *Dynamic Community Detection*, pages 1–10. 01 2017.

[18] R. Cazabet, H. Takeda, M. Hamasaki, and F. Amblard. Using dynamic community detection to identify trends in user-generated content. *Social Network Analysis and Mining*, 2(4):361–371, 2012.

[19] D. Chakrabarti, R. Kumar, and A. Tomkins. Evolutionary clustering. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 554–560. ACM, 2006.

[20] J. Chan, J. Bailey, and C. Leckie. Discovering correlated spatio-temporal changes in evolving graphs. *Knowledge and Information Systems*, 16(1):53–96, 2008.

[21] G. Chen, Y. Wang, and J. Wei. A new multiobjective evolutionary algorithm for community detection in dynamic complex networks. *Mathematical problems in engineering*, 2013, 2013.

[22] H. Cheng and S. Yang. Genetic algorithms with immigrants schemes for dynamic multicast problems in mobile ad hoc networks. *Engineering Applications of Artificial Intelligence*, 23(5):806–819, 2010.

[23] A. Clauset, M. E. Newman, and C. Moore. Finding community structure in very large networks. *Physical review E*, 70(6):066111, 2004.

[24] H. G. Cobb and J. J. Grefenstette. Genetic algorithms for tracking changing environments. Technical report, Naval Research Lab Washington DC, 1993.

[25] G. Cordasco and L. Gargano. Community detection via semi-synchronous label propagation algorithms. In *2010 IEEE international workshop on: business applications of social network analysis (BASNA)*, pages 1–8. IEEE, 2010.

[26] N. Dakiche, F. B.-S. Tayeb, Y. Slimani, and K. Benatchba. Tracking community evolution in social networks: A survey. *Information Processing & Management*, 56(3):1084–1102, 2019.

[27] K. Deb and H. Jain. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: solving problems with box constraints. *IEEE transactions on evolutionary computation*, 18(4):577–601, 2013.

[28] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002.

[29] J. Derrac, S. García, D. Molina, and F. Herrera. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation*, 1(1):3–18, 2011.

[30] D. Duan, Y. Li, Y. Jin, and Z. Lu. Community mining on dynamic weighted directed graphs. In *Proceedings of the 1st ACM international workshop on Complex networks meet information & knowledge management*, pages 11–18. ACM, 2009.

[31] T. Falkowski, A. Barth, and M. Spiliopoulou. Studying community dynamics with an incremental graph mining algorithm. In *Americas Conference on Information Systems (AMCIS)*, page 29, 2008.

[32] G. W. Flake, S. Lawrence, and C. L. Giles. Efficient identification of web communities. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 150–160, 2000.

[33] F. Folino and C. Pizzuti. Multiobjective evolutionary community detection for dynamic networks. In *GECCO*, pages 535–536, 2010.

[34] F. Folino and C. Pizzuti. An evolutionary multiobjective approach for community discovery in dynamic networks. *IEEE Transactions on Knowledge and Data Engineering*, 26(8):1838–1852, 2013.

[35] F. Folino and C. Pizzuti. An evolutionary multiobjective approach for community discovery in dynamic networks. *IEEE Transactions on Knowledge and Data Engineering*, 26(8):1838–1852, 2014.

[36] S. Fortunato. Community detection in graphs. *Physics reports*, 486(3-5):75–174, 2010.

[37] C.-K. Goh and K. C. Tan. A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 13(1):103–127, 2008.

[38] R. Görke, P. Maillard, A. Schumm, C. Staudt, and D. Wagner. Dynamic graph clustering combining modularity and smoothness. *Journal of Experimental Algorithmics (JEA)*, 18:1–5, 2013.

[39] D. Greene, D. Doyle, and P. Cunningham. Tracking the evolution of communities in dynamic social networks. In *2010 International Conference on Advances in Social Networks Analysis and Mining*, pages 176–183. IEEE, 2010.

[40] A. Gupta, Y.-S. Ong, and L. Feng. Multifactorial evolution: toward evolutionary multitasking. *IEEE Transactions on Evolutionary Computation*, 20(3):343–357, 2015.

[41] A. Hajibagheri, G. Sukthankar, K. Lakkaraju, H. Alvari, R. T. Wigand, and N. Agarwal. Using massively multiplayer online game data to analyze the dynamics of social interactions. *Social Interactions in Virtual Worlds: An Interdisciplinary Perspective*, 2018.

[42] J. Han, W. Li, L. Zhao, Z. Su, Y. Zou, and W. Deng. Community detection in dynamic networks via adaptive label propagation. *PloS one*, 12(11):e0188655, 2017.

[43] A.-S. Himmel, H. Molter, R. Niedermeier, and M. Sorge. Enumerating maximal cliques in temporal graphs. In *Proceedings of the International Conference on Advances in Social Networks Analysis and Mining*, pages 337–344, 2016.

[44] J. H. Holland. Genetic algorithms and the optimal allocation of trials. *SIAM journal on computing*, 2(2):88–105, 1973.

[45] P. W. Holland, K. B. Laskey, and S. Leinhardt. Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137, 1983.

[46] Y. Hulovatyy and T. Milenković. Scout: simultaneous time segmentation and community detection in dynamic networks. *Scientific reports*, 6:37557, 2016.

[47] R. Kannan, S. Vempala, and A. Vetta. On clusterings: Good, bad and spectral. *Journal of the ACM (JACM)*, 51(3):497–515, 2004.

[48] B. Karrer, E. Levina, and M. E. Newman. Robustness of community structure in networks. *Physical review E*, 77(4):046119, 2008.

[49] J. Kelsey and J. Timmis. Immune inspired somatic contiguous hypermutation for function optimisation. In *Genetic and Evolutionary Computation Conference*, pages 207–218. Springer, 2003.

[50] W. H. Kruskal and W. A. Wallis. Use of ranks in one-criterion variance analysis. *Journal of the American statistical Association*, 47(260):583–621, 1952.

[51] A. Lancichinetti and S. Fortunato. Limits of modularity maximization in community detection. *Physical review E*, 84(6):066122, 2011.

[52] A. Lancichinetti, S. Fortunato, and J. Kertész. Detecting the overlapping and hierarchical community structure in complex networks. *New journal of physics*, 11(3):033015, 2009.

[53] A. Lancichinetti, S. Fortunato, and F. Radicchi. Benchmark graphs for testing community detection algorithms. *Physical review E*, 78(4):046110, 2008.

[54] C. Largeron, P.-N. Mougel, O. Benyahia, and O. R. Zaïane. Dancer: dynamic attributed networks with community structure generation. *Knowledge and Information Systems*, 53(1):109–151, 2017.

[55] J. Li, L. Huang, T. Bai, Z. Wang, and H. Chen. Cdbia: a dynamic community detection method based on incremental analysis. In *2012 International Conference on Systems and Informatics (ICSAI2012)*, pages 2224–2228. IEEE, 2012.

[56] R.-H. Li, J. Su, L. Qin, J. X. Yu, and Q. Dai. Persistent community search in temporal networks. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, pages 797–808. IEEE, 2018.

[57] Y.-R. Lin, Y. Chi, S. Zhu, H. Sundaram, and B. L. Tseng. Analyzing communities and their evolutions in dynamic social networks. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 3(2):8, 2009.

[58] Y.-R. Lin, J. Sun, H. Sundaram, A. Kelliher, P. Castro, and R. Konuru. Community discovery via metagraph factorization. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 5(3):17, 2011.

[59] F. Liu, D. Choi, L. Xie, and K. Roeder. Global spectral clustering in dynamic networks. *Proceedings of the National Academy of Sciences*, 115(5):927–932, 2018.

[60] M. López-Ibánez, L. Paquete, and T. Stützle. Exploratory analysis of stochastic local search algorithms in biobjective optimization. In *Experimental methods for the analysis of optimization algorithms*, pages 209–222. Springer, 2010.

[61] J. Ma, J. Liu, W. Ma, M. Gong, and L. Jiao. Decomposition-based multiobjective evolutionary algorithm for community detection in dynamic social networks. *The Scientific World Journal*, 2014, 2014.

[62] B. Mitra, L. Tabourier, and C. Roth. Intrinsically dynamic network communities. *Computer Networks*, 56(3):1041–1053, 2012.

[63] A. Miyauchi and Y. Kawase. Z-score-based modularity for community detection in networks. *PloS one*, 11(1):e0147805, 2016.

[64] N. Mori, H. Kita, and Y. Nishikawa. Adaptation to a changing environment by means of the feedback thermodynamical genetic algorithm. In *International Conference on Parallel Problem Solving from Nature*, pages 149–158. Springer, 1998.

[65] P. J. Mucha, T. Richardson, K. Macon, M. A. Porter, and J.-P. Onnela. Community structure in time-dependent, multiscale, and multiplex networks. *science*, 328(5980):876–878, 2010.

[66] A. J. Nebro, J. J. Durillo, F. Luna, B. Dorronsoro, and E. Alba. Mocell: A cellular genetic algorithm for multiobjective optimization. *International Journal of Intelligent Systems*, 24(7):726–746, 2009.

[67] M. E. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

[68] K. P. Ng and K. C. Wong. A new diploid scheme and dominance change mechanism for non-stationary function optimization. In *Proceedings of the 6th international conference on genetic algorithms*, pages 159–166, 1995.

[69] N. P. Nguyen, T. N. Dinh, Y. Xuan, and M. T. Thai. Adaptive algorithms for detecting community structure in dynamic social networks. In *2011 Proceedings IEEE INFOCOM*, pages 2282–2290. IEEE, 2011.

[70] T. T. Nguyen. *Continuous dynamic optimisation using evolutionary algorithms*. PhD thesis, University of Birmingham, 2011.

[71] T. T. Nguyen, S. Yang, and J. Branke. Evolutionary dynamic optimization: A survey of the state of the art. *Swarm and Evolutionary Computation*, 6:1–24, 2012.

[72] E. Osaba, J. Del Ser, D. Camacho, M. N. Bilbao, and X.-S. Yang. Community detection in networks using bio-inspired optimization: Latest developments, new results and perspectives with a selection of recent meta-heuristics. *Applied Soft Computing*, 87:106010, 2020.

[73] G. Palla, A.-L. Barabási, and T. Vicsek. Quantifying social group evolution. *Nature*, 446(7136):664, 2007.

[74] A. Panizo, G. Bello-Orgaz, and D. Camacho. A genetic algorithm with local search based on label propagation for detecting dynamic communities. In *International Symposium on Intelligent and Distributed Computing*, pages 319–328. Springer, 2018.

[75] A. Panizo, G. Bello-Orgaz, A. Ortega, and D. Camacho. Community finding in dynamic networks using a genetic algorithm improved via a hybrid immigrants scheme. In *Data Science and Knowledge Engineering for Sensing Decision Support: Proceedings of the 13th International FLINS Conference (FLINS 2018)*, pages 591–598. World Scientific, 2018.

[76] A. Panizo-LLedot, G. Bello-Orgaz, and D. Camacho. A multi-objective genetic algorithm for detecting dynamic communities using a local search driven immigrant's scheme. *Future Generation Computer Systems*, pages 1–16, 2019.

[77] A. Panizo-LLedot, G. Bello-Orgaz, and D. Camacho. A multi-objective genetic algorithm for detecting dynamic communities using a local search driven immigrant's scheme. *Future Generation Computer Systems*, 110:960–975, 2020.

[78] A. Panizo-LLedot, M. Pedemonte, G. Bello-Orgaz, and D. Camacho. Addressing evolutionary-based dynamic problems: a new methodology for evaluating immigrants strategies in mogas. *IEEE Access*, 2022.

[79] L. Peel and A. Clauset. Detecting change points in the large-scale structure of evolving networks. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[80] C. Pizzuti. Ga-net: A genetic algorithm for community detection in social networks. In *International conference on parallel problem solving from nature*, pages 1081–1090. Springer, 2008.

[81] C. Pizzuti. Evolutionary computation for community detection in networks: A review. *IEEE Transactions on Evolutionary Computation*, 22(3):464–483, 2017.

[82] R. Poli, J. Kennedy, and T. Blackwell. Particle swarm optimization. *Swarm intelligence*, 1(1):33–57, 2007.

[83] H. Qin, R.-H. Li, G. Wang, L. Qin, Y. Cheng, and Y. Yuan. Mining periodic cliques in temporal networks. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, pages 1130–1141. IEEE, 2019.

[84] M. Riekert, K. Malan, and A. Engelbrect. Adaptive genetic programming for dynamic classification problems. In *2009 IEEE congress on evolutionary computation*, pages 674–681. IEEE, 2009.

[85] G. Rossetti and R. Cazabet. Community discovery in dynamic networks: a survey. *ACM Computing Surveys (CSUR)*, 51(2):35, 2018.

[86] G. Rossetti, L. Pappalardo, D. Pedreschi, and F. Giannotti. Tiles: an online algorithm for community discovery in dynamic social networks. *Machine Learning*, 106(8):1213–1241, 2017.

[87] M. Rosvall and C. T. Bergstrom. Maps of information flow reveal community structure in complex networks. *arXiv preprint physics.soc-ph/0707.0609*, 2007.

[88] M. Rosvall and C. T. Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 105(4):1118–1123, 2008.

[89] A. Sanfeliu and K.-S. Fu. A distance measure between attributed relational graphs for pattern recognition. *IEEE transactions on systems, man, and cybernetics*, (3):353–362, 1983.

[90] S. E. Schaeffer. Graph clustering. *Computer science review*, 1(1):27–64, 2007.

[91] C. E. Shannon. A mathematical theory of communication. *ACM SIGMOBILE mobile computing and communications review*, 5(1):3–55, 2001.

[92] C. Shi, P. S. Yu, Z. Yan, Y. Huang, and B. Wang. Comparison and selection of objective functions in multiobjective community detection. *Computational Intelligence*, 30(3):562–582, 2014.

[93] J. Sun, C. Faloutsos, C. Faloutsos, S. Papadimitriou, and P. S. Yu. Graphscope: parameter-free mining of large time-evolving graphs. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 687–696. ACM, 2007.

[94] E. G. Tajeuna, M. Bouguessa, and S. Wang. Tracking the evolution of community structures in time-evolving social networks. In *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 1–10. IEEE, 2015.

[95] E. G. Tajeuna, M. Bouguessa, and S. Wang. Modeling and predicting community structure changes in time-evolving social networks. *IEEE Transactions on Knowledge and Data Engineering*, 31(6):1166–1180, 2018.

[96] L. Tang, H. Liu, J. Zhang, and Z. Nazeri. Community evolution in dynamic multi-mode networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 677–685. ACM, 2008.

[97] C. Tantipathananandh, T. Berger-Wolf, and D. Kempe. A framework for community identification in dynamic social networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 717–726. ACM, 2007.

[98] R. Tinós and S. Yang. A self-organizing random immigrants genetic algorithm for dynamic optimization problems. *Genetic Programming and Evolvable Machines*, 8(3):255–286, 2007.

[99] V. A. Traag, R. Aldecoa, and J.-C. Delvenne. Detecting communities using asymptotical surprise. *Physical review e*, 92(2):022816, 2015.

[100] R. K. Ursem. Multinational ga optimization techniques in dynamics environments. In *Genetic and Evolutionary Computation Conference*, pages 19–26, 2000.

[101] T. Viard, M. Latapy, and C. Magnien. Computing maximal cliques in link streams. *Theoretical Computer Science*, 609:245–252, 2016.

[102] T. Viard, C. Magnien, and M. Latapy. Enumerating maximal cliques in link streams with durations. *Information Processing Letters*, 133:44–48, 2018.

[103] Y. Wang and M. Wineberg. Estimation of evolvability genetic algorithm and dynamic environments. *Genetic Programming and Evolvable Machines*, 7(4):355, 2006.

[104] Z. Wang, Z. Li, G. Yuan, Y. Sun, X. Rui, and X. Xiang. Tracking the evolution of overlapping communities in dynamic social networks. *Knowledge-Based Systems*, 157:81–97, 2018.

[105] S. Wasserman, K. Faust, et al. Social network analysis: Methods and applications. 1994.

[106] K. S. Xu and A. O. Hero. Dynamic stochastic blockmodels for time-evolving social networks. *IEEE Journal of Selected Topics in Signal Processing*, 8(4):552–562, 2014.

[107] S. Yang. Memory-based immigrants for genetic algorithms in dynamic environments. In *Proceedings of the 7th annual conference on Genetic and evolutionary computation*, pages 1115–1122, 2005.

[108] S. Yang. On the design of diploid genetic algorithms for problem optimization in dynamic environments. In *2006 IEEE International Conference on Evolutionary Computation*, pages 1362–1369. IEEE, 2006.

[109] S. Yang. Genetic algorithms with elitism-based immigrants for changing optimization problems. In *Workshops on Applications of Evolutionary Computation*, pages 627–636. Springer, 2007.

[110] S. Yang. Genetic algorithms with memory-and elitism-based immigrants in dynamic environments. *Evolutionary Computation*, 16(3):385–416, 2008.

[111] S. Yang and R. Tinós. A hybrid immigrants scheme for genetic algorithms in dynamic environments. *International Journal of Automation and Computing*, 4(3):243–254, 2007.

[112] T. Yang, Y. Chi, S. Zhu, Y. Gong, and R. Jin. Detecting communities and their evolutions in dynamic social networks—a bayesian approach. *Machine learning*, 82(2):157–189, 2011.

[113] X.-S. Yang. Bat algorithm for multi-objective optimisation. *International Journal of Bio-Inspired Computation*, 3(5):267–274, 2011.

[114] Y. Yin, Y. Zhao, H. Li, and X. Dong. Multi-objective evolutionary clustering for large-scale dynamic community detection. *Information Sciences*, 549:269–287, 2021.

[115] E. Yu and P. N. Suganthan. Evolutionary programming with ensemble of explicit memories for dynamic optimization. In *2009 IEEE Congress on Evolutionary Computation*, pages 431–438. IEEE, 2009.

[116] X. Yu, Y. Jin, K. Tang, and X. Yao. Robust optimization over time—a new perspective on dynamic optimization problems. In *IEEE Congress on evolutionary computation*, pages 1–6. IEEE, 2010.

[117] X. Yu, K. Tang, T. Chen, and X. Yao. Empirical analysis of evolutionary algorithms with immigrants schemes for dynamic optimization. *Memetic Computing*, 1(1):3–24, 2009.

[118] A. Zakrzewska and D. A. Bader. A dynamic algorithm for local community detection in graphs. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 559–564. ACM, 2015.

[119] Q. Zhang and H. Li. Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on evolutionary computation*, 11(6):712–731, 2007.

[120] S. Zhang, X.-M. Ning, C. Ding, and X.-S. Zhang. Determining modular organization of protein interaction networks by maximizing modularity density. In *BMC systems biology*, volume 4, pages 1–12. BioMed Central, 2010.

[121] X. Zhou, X. Zhao, and Y. Liu. A multiobjective discrete bat algorithm for community detection in dynamic networks. *Applied Intelligence*, 48(9):3081–3093, 2018.

[122] E. Zitzler and S. Künzli. Indicator-based selection in multiobjective search. In *International conference on parallel problem solving from nature*, pages 832–842. Springer, 2004.

[123] E. Zitzler, M. Laumanns, and L. Thiele. Spea2: Improving the strength pareto evolutionary algorithm. *TIK-report*, 103, 2001.

[124] E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE transactions on Evolutionary Computation*, 3(4):257–271, 1999.