A simple CSP-based model for Unmanned Air Vehicle Mission Planning

Cristian Ramirez-Atencia*, Gema Bello-Orgaz*, Maria D. R-Moreno† and David Camacho*
*Departamento de Ingeniería Informática,
Universidad Autónoma de Madrid,
C/Francisco Tomás y Valiente 11, 28049 Madrid, Spain
Emails: cristian.ramirez@inv.uam.es, {gema.bello, david.camacho}@uam.es
†Departamento de Automática,
Universidad de Alcalá,
Carretera Madrid Barcelona, km 33 600, 28871 Madrid, Spain
Email: mdolores@aut.uah.es

***Abstract*—The problem of Mission Planning for a large number of Unmanned Air Vehicles (UAV) can be formulated as a Temporal Constraint Satisfaction Problem (TCSP). It consists on a set of locations that should visit in different time windows, and the actions that the vehicle can perform based on its features such as the payload, speed or fuel capacity. In this paper, a temporal constraint model is implemented and tested by performing Backtracking search in several missions where its complexity has been incrementally modified. The experimental phase consists on two different phases. On the one hand, several mission simulations containing (n) UAVs using different sensors and characteristics located in different waypoints, and (m) requested tasks varying mission priorities have been carried out. On the other hand, the second experimental phase uses a backtracking algorithm to look through the whole solutions space to measure the scalability of the problem. This scalability has been measured as a relation between the number of tasks to be performed in the mission and the number of UAVs needed to perform it.**

***Keywords*—Unmanned Aircraft Systems, Mission Planning, Temporal Constraint Satisfaction Problems, Backtracking**

I. Introduction

Unmanned Aircraft Systems (UAS) can take advantage of planning techniques where the application domain can be defined as the process of generating tactical goals for a team of Unmanned Air Vehicles (UAVs). Nowadays, these vehicles are controlled remotely from ground control stations by humans operators who use legacy mission planning systems.

Mission planning for UAS can be defined as the process of planning the locations to visit (waypoints) and the actions that the vehicle can perform (loading/dropping a load, taking videos/pictures, acquiring information), typically over a time period. These planning problems can be solved using different methods such as Mixed-Integer Lineal Programming (MILP) [1], Simulated Annealing [2], Auction algorithms [3], etc. Usually, these methods are the best way to find the optimal solutions but, as the number of restrictions increase, the complexity grows exponentially because it is a NP-hard problem.

Other modern approaches formulate the mission planning problem as a Constraint Satisfaction Problem (CSP) [4], where the tactic mission is modelled and solved using constraint satisfaction techniques. Our work will deal with multiple UAVs that must perform one or more tasks in a set of waypoints and in specific time windows. The solution plans obtained should fulfill all the constraints given by the different components and capabilities of the UAVs involved over the time periods given. Therefore a Temporal Constraint Satisfaction Problem (TCSP) representation is needed.

There are several functional CSP modellers and solvers, such as Gecode [5], Opturion-CPX [6] or Choco [7], among others. Some of them are pretty efficient, such as Gecode, which has been used in many research projects in last the years [8] [9]. In this paper, we model a mission planning problem as a TCSP for a team of UAVs, using Gecode as the CSP modeller to program the constraints of the problem, and solve them using Backtracking (BT) search. The scalability of the problem is analysed as the number of tasks and UAVs increase, paying special attention to whether tasks collide in time or not.

The rest of the paper is structured as follows: section II shows the state of the art in the aforementioned topics of Mission Planing and CSPs. Section III describes how a Misison is defined in the UAV domain. Section IV is focused on the modellization of the problem as a TCSP. Sections V and VI explains the experiments performed and the experimental results obtained. Finally, the last section presents the final analysis and conclusions of this work.

II. Related Work

A. Mission Planning

Planning has been an area of research in Artificial Intelligence (AI) for over three decades. A variety of tasks including robotics [10][11], web-based information gathering [12][13][14], autonomous agents [15][16][17] and mission control [18] have benefited from planning techniques.

In the literature there are some attempts to implement UAS guidance systems that achieve mission planning and decision making. Doherty [19] presents an architectural framework for mission planning and execution monitoring, using temporal action logic (TAL) for reasoning about actions and changes; and its integration into a fully deployed unmanned helicopter.

A similar project, called ReSSAC (Search and Rescue by Cooperative Autonomous System), was carried out by

the French Aerospace Lab (ONERA) for search and rescue scenarios [20]. The problem was modelled using the Markov Decision Process (MDP) framework and dynamic programming algorithms for the mission planning. Konigsbuch [21] extends this model and integrates it in a robotic helicopter.

Finally, German Aerospace Centre (DLR) also developed a mission management system based on the behavior paradigm [22] which has been integrated onboard the ARTIS helicopter and validated in different scenarios, including waypoints following and search and track missions.

An essential concept in Mission Planning is cooperation or collaboration, which occurs at a higher level when various UASs work together in a common mission sharing data and controlling actions together. Besides, techniques and algorithms for cooperative missions can be divided into two main categories: cooperative perception and cooperative mission planning and decision-making [23].

Regarding cooperative mission planning, there are few contributions that deal with multi UAS problems in a deliberative paradigm (cooperative task assignment and mission planning). A mission planner should provide a list of assignment tasks where each task is assigned to an available vehicle that should perform this task. This assignment is based on information about the tasks and the capabilities of the vehicles. Bethke et al. [24] propose an algorithm for cooperative task assignment that extends the receding-horizon task assignment (RHTA) algorithm [25] developed at MIT. The modified RHTS algorithm solves an optimization problem to select the optimal sequence of tasks for each UAS. Another approach by Kvarnstrom et al. [26] propose a new mission planning algorithm for collaborative UAS based on combining ideas from forward-chaining planning with partial-order planning, leading to a new hybrid partial-order forward-chaining (POFC) framework that meets the requirements on centralization, abstraction, and distribution found in realistic emergency services settings.

B. Constraint Satisfaction Problems

A mission can be described as a set of goals that are achieved by performing some task with a group of resources over a period of time. The whole problem can be summed up in finding the correct schedule of resource-task assignments that satisfies the proposed constraints, like a CSP that can be defined as [27]:

- A set of variables $V = v_1, , v_n$
- for each variable, a finite set of possible values D_i (its domain)
- and a set of constraints C_i restricting the values that variables can simultaneously take

In a CSP, the states are defined by the values of the variables and the goal test specifies the constraints that the values must obey. Many kind of designs and scheduling problems can be expressed as CSPs.

Typically, a CSP is represented as a graph, with the pair <Variables,Values> in the nodes and the constraints in the edges, although there are other representations as those presented in [28][29][30] for Ant Colony Optimization and videogames. There are many studied methods to search the space of solutions for CSPs, such as BT, Backjumping (BJ) or look-ahead techniques (i.e. Forward Checking (FC) [31]). These algorithms are usually combined with other techniques like consistency techniques [32] (domain consistency, arc consistency or path consistency) to modify the CSP and ensure its local consistency conditions.

BT [33] is a method of solving CSP by incrementally extending a partial solution that specifies consistent values for some of the variables, towards a complete solution, and by repeatedly choosing a value for another variable consistent with the values in the current partial solution. If a partial solution violates any of the constraints, backtracking is performed to the most recently instantiated variable that still has alternatives available. BT is strictly better than random generate-and-test algorithm, however, its running complexity for most nontrivial problems is exponential.

A TCSP is a particular class of CSP where variables represent times (time points, time intervals or durations) and constraints represent sets of allowed temporal relations between them [34]. Different classes of constraints are characterized by the underlying set of basic temporal relations (BTR). Most types of TCSPs can be represented with Point Algebra (PA), with BTR = $\{\varnothing, <, =, >, \leq, \geq, ?\}$.

In the related literature, Mouhoub [35] proved that on real-time or Maximal TCSPs (MTCSPs), the best methods for solving them were Min-Conflict-Random-Walk (MCRW) in the case of under-constrained and middle-constrained problems, and Tabu search and Steepest-Descent-Random-Walk (SDRW) in the over-constrained case. He also developed a temporal model, TemPro [36], which was based on interval algebra, to translate an application involving temporal information into a CSP.

A TCSP can perfectly represent an UAS mission as a set of temporal constraints over the time the tasks in the mission start and end. Besides the temporal constraints, the problem has various constraints imposing the proficiency of the UAVs to perform the tasks.

III. Defining UAV Mission Plans

A UAS mission can be defined as a number n of tasks to accomplish for a team of UAVs. A task could be exploring a specific area or search for an object in a zone, which can be carry out thanks to the sensors belonging to a particular UAV as can be seen in Table I. Each task must be performed in a specific geographic *area*, in a specific *time interval* and needs an amount of *payloads* to be accomplished.

TABLE I: Different task actions considered

Action	Payload Needed
Taking pictures of a zone	• Camera EO/IR
Taking real-time pictures of a zone	• Camera EO/IR • Communications Equipment
Tracking a zone	• Radar SAR

To perform a mission, there are a number m of UAVs, each one with some specific characteristics:

- The fuel consumption rate
- The maximum reachable speed
- The minimum cruise speed
- The maximum and minimum flight altitude
- Whether it has or not permission to go to restricted areas
- An amount of capacities or payloads (cameras, radars, communication equipments, ...) available.

Moreover, in each point in time, each UAV is positioned at some *coordinates* and is filled with an amount of *fuel*.

Therefore, the main goal to solve the problem is to assign each task with an UAV that is able to perform it, and a start time of the UAV departure to reach the task area in time. Note that the UAV could be parked at an airport or in flight after performing a previous task. For this simple approach, we will despise the UAV fuel and time costs due to the takeoff and loiter during tasks development. Figure 1 shows an overview of the mission planning process.

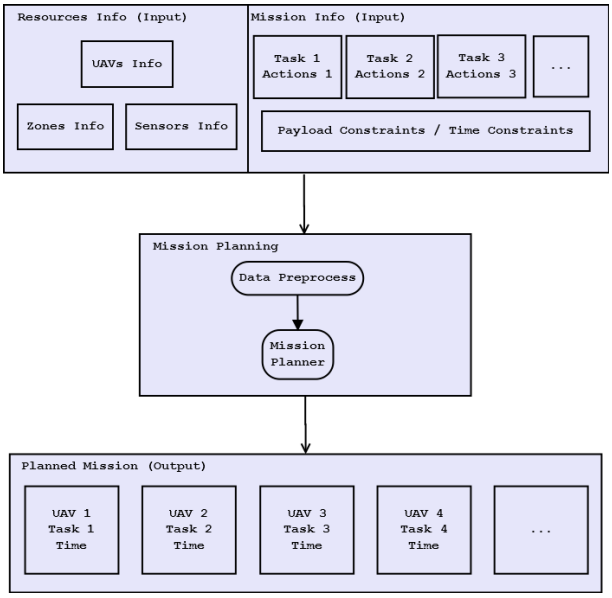


Fig. 1: Mission Planning overview.

IV. Modelling UAV Missions as a TCSP

Given the previous definition of mission planning, it is pretty simple to note that it can be formulated as a scheduling problem with constraints, including particularly temporal constraints related to the UAV departure times the tasks have to be performed.

In this approach, the problem domain is modelled as a TCSP where the main variables are the *tasks* and their values will be the *UAVs* that perform each task and their respective *departure times*. Moreover, there are two additional variables, the *fuel cost* and *distance travelled* for each task, that can be deduced from tasks assignment and UAV characteristics. The different constraints defined to model this approach are as follows:

- Temporal constraints assuring an UAV does not perform two tasks at the same time. Let t_i be the time the i^{th} task is completed, and τ_i the duration of the task, i.e. the difference between the end and the start (not to be confused with the departure) times of the task. Besides, let k be an UAV that executes two tasks i and j, where i takes place before j, then t_i must precede the departure time of k for j. Here, we also define the distance travelled by k to reach a task i in time, $d_{k\to i}$, and the mean cruise speed the UAV flies to reach that task, $v_{k\to i}$. Then the following inequality must be obeyed:

$$t_i \leqslant t_j - \frac{d_{k\to j}}{v_{k\to j}} - \tau_j \tag{1}$$

To compute the distance, we need to know where the UAV is located before the start of the task. For this purpose, we have created a $m \times n$ matrix pos of task to UAV positions, with $pos_{k,i}$ denoting the position of vehicle k before the start of task i. Each time an assignment is done, this matrix is updated changing the row of the assigned UAV, so any subsequent to the task assigned would be the middle point of the area where the task is developed. With this matrix, we can compute every distance between two points using the Haversine formula with the latitude and longitude

$$d_{2D} = 2r_{EARTH} \arcsin\left(\sqrt{\sin^2\left(\frac{lat_2 - lat_1}{2}\right) + \cos(lat_1)\cos(lat_2)\sin^2\left(\frac{long_2 - long_1}{2}\right)}\right) \tag{2}$$

and the Euclidean distance with the altitude

$$d_{3D} = \sqrt{d^2 + (alt_2 - alt_1)^2}. \tag{3}$$

Therefore, if the distance from the position of the UAV can be computed as the middle point of the area of the task, we can conclude that:

$$d_{k\to i} = i.area.distance(pos_{k,i}) \tag{4}$$

- Speed window constraints: the mean cruise speed of the UAV k necessary to perform the task i depends on the speed window $v_{k,max}$ and $v_{k,min}$ by:

$$v_{k,min} \leq v_{k\to i} \leq v_{k,max} \tag{5}$$

- Payload constraints: another constraint is whether an UAV carries the corresponding payload to perform a task. Let P_k denote the payloads available for UAV k and P_i the payloads needed for the task i (performed by k), then:

$$P_i \subseteq P_k, \tag{6}$$

- Altitude window constraints: an UAV k, with an altitude window $k_{h_{max}}$ and $k_{h_{min}}$, performing a task i developed in an area with an altitude window h_{max} and h_{min}, must obey:

$$k.h_{max} \geq i.area.h_{max} \quad (7)$$
$$k.h_{min} \leq i.area.h_{min} \quad (8)$$

- Zone permission constraints: another constraint is the implication that a restricted area has in the tasks to perform. Just UAVs with permissions in those areas shall perform the tasks.
- Fuel constraints: Finally, we must constraint the fuel cost for each UAV. The fuel cost for an UAV k performing a task i is $f_i = k.fuelConsume * (d_{k \to i} + \tau_i \bar{v_i})$, being $\bar{v_i}$ the speed at which the task is performed. It obeys the following inequality:

$$\sum_{i \in T_k} f_i \leqslant k.fuel \quad (9)$$

V. Model Implementation

Using Gecode, we have modelled the problem explained in the previous section. Then, we have designed 10 missions, each one composed by an increasing number of tasks from 1 to 10, i.e the first mission has one task; the second, two tasks; and so on. Table II shows the 10 considered tasks, where the first mission will execute task with ID 1; the second will execute tasks with IDs 1 and 2; and so on. This table shows the duration of the tasks instead of the start and end times. These times will be fixed on the next section depending on the number of dependencies between the tasks.

TABLE II: UAS mission with 10 tasks

Task ID	Actions	Duration	Zone altitude window	Restricted zone?
1	Taking pictures of a zone	25 min	$[1.5 - 5]$ km	NO
2	Tracking a zone	20 min	$[1.5 - 5]$ km	NO
3	Taking real-time pictures of a zone	30 min	$[2.5 - 6.15]$ km	YES
4	Taking pictures of a zone	25 min	$[0.5 - 3.75]$ km	NO
5	Tracking a zone	35 min	$[0.5 - 3.75]$ km	NO
6	Taking real-time pictures of a zone	30 min	$[3.85 - 5]$ km	NO
7	Taking real-time pictures of a zone	25 min	$[3.85 - 5]$ km	NO
8	Taking real-time pictures of a zone	12 min	$[1.5 - 5]$ km	NO
9	Taking pictures of a zone	20 min	$[1.5 - 5]$ km	NO
10	Tracking a zone	25 min	$[2.5 - 6.15]$ km	YES

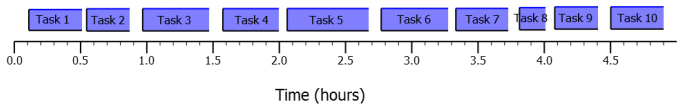
Different scenarios for solving the missions have been prepared with an increasing number of UAVs able to perform the tasks. The tasks contain several constraints, so when the number of tasks is very high, a high number of UAVs is also needed, mainly because of the fuel constraints. We have considered missions with 4 to 7 vehicles available to perform the tasks (see Table III). For a mission with 4 vehicles, we use UAVs with IDs 1 to 4; for a mission with 5 vehicles, UAVs with IDs 1 to 5; and so on.

Then, each scenario has been implemented with different perspectives based on the time dependencies between the tasks. Figure 2a shows an scenario with no time dependencies between tasks, i.e. the tasks do not collide in time. Figure 2b shows an scenario where each task collides in time with the previous task, i.e. there are $n-1$ dependencies, with n the number of tasks. Finally, when each task collides in time with the two previous tasks, i.e. there are $2(n-1)-1$ dependencies, we have the scenario shown in Figure 2c.

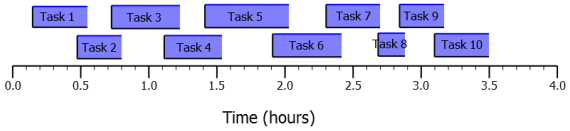
TABLE III: Available UAVs

UAV ID	Cruise speed window	Altitude window	Restricted zone permission	Fuel consume	Initial Fuel
1	$[90 - 110]$ km/h	$[0.3 - 6.5]$ km	YES	0.159 L/km	97.52 L
2	$[90 - 110]$ km/h	$[0.3 - 6]$ km	NO	0.159 L/km	58.48 L
3	$[110 - 190]$ km/h	$[0.8 - 10]$ km	YES	0.2 L/km	140.23 L
4	$[90 - 110]$ km/h	$[0.3 - 6]$ km	YES	0.159 L/km	47.12 L
5	$[90 - 110]$ km/h	$[0.3 - 6]$ km	NO	0.159 L/km	101.48 L
6	$[90 - 110]$ km/h	$[0.3 - 6]$ km	NO	0.159 L/km	101.37 L
7	$[90 - 110]$ km/h	$[0.3 - 6]$ km	NO	0.159 L/km	58.15 L

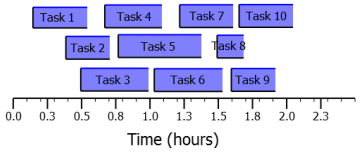
UAV ID	Payloads available
1	• Camera EO/IR • Radar SAR • Communications Equipment
2	• Camera EO/IR
3	• Camera EO/IR • Radar SAR
4	• Camera EO/IR
5	• Camera EO/IR • Radar SAR • Communications Equipment
6	• Camera EO/IR • Radar SAR • Communications Equipment
7	• Camera EO/IR



(a) No dependencies



(b) Dependency of each task with the previous task.



(c) Dependency of each task with the two previous tasks.

Fig. 2: Three perspectives of the scenarios based on the number of time dependencies between the tasks.

VI. Experimental Results

BT search implemented by Gecode solver has been used to solve the missions explained in the previous section, analysing the runtime spent in the process. This search algorithm performs constraint propagation with different consistency levels depending on the type of the constraint. For all the developed constraints in our problem, domain consistency is applied.

Figures 3 and 4 shows the number of solutions and runtime obtained when the tasks do not collide in time (see Figure 2a). As we can see, the growth of the number of solutions is nearly exponential as the number of tasks increase. Indeed, the exponentiality is higher and more appreciable as the number of UAVs increase. For the runtime, the situation is similar, and the exponentiality growth is much higher. So it is clear that the scalability of the problem as the number of variables increase is exponential.

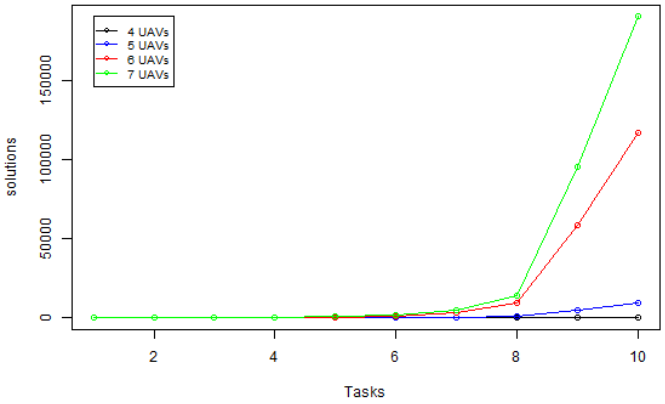


Fig. 3: Number of solutions for missions with 1 to 10 tasks, where each task has no dependencies with any other, for groups of 4 to 7 UAVs.

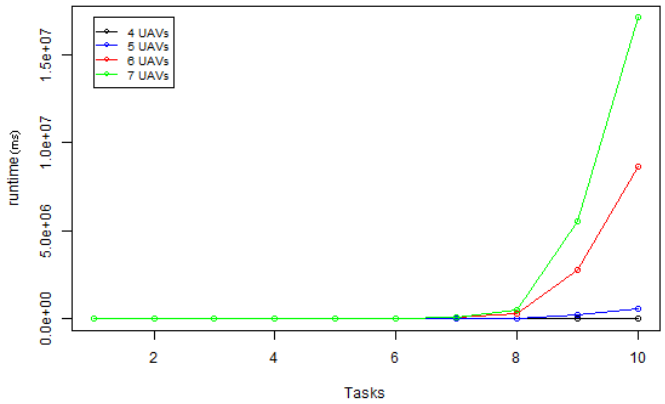


Fig. 4: Runtime for missions with 1 to 10 tasks, where each task has no dependencies with any other, for groups of 4 to 7 UAVs.

On the other hand, Figures 5 and 6 shows what happens when each task collides in time with the previous task (see Figure 2b). As it can be seen, the growth is still pretty exponential for both the number of solutions and the runtime, but much smaller than with no dependencies. We also note that for less UAVs to perform the tasks, the exponentiality of the number of solutions disappears. This is because of the high number of constraints that, in conjunction with the new temporal constraints due to the tasks dependencies, reduces the space search and, for a high number of tasks, makes the problem highly complex.

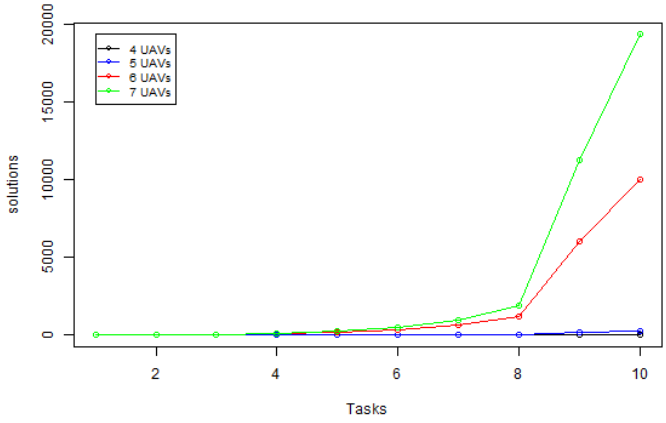


Fig. 5: Number of solutions for missions with 1 to 10 tasks, where each task collides in time with the previous, for groups of 4 to 7 UAVs.

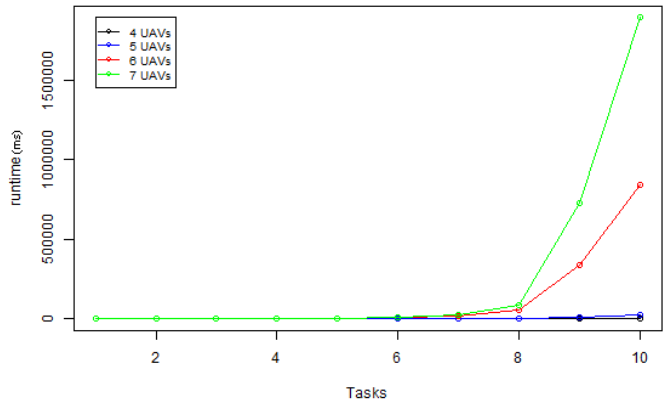


Fig. 6: Runtime for missions with 1 to 10 tasks, where each task collides in time with the previous, for groups of 4 to 7 UAVs.

When each task collides in time with the two previous tasks (see Figure 2c), the results in Figures 7 and 8 show that the growth of the runtime is still exponential, but much smaller than in the two previous cases. On the other hand, the growth of the number of solutions has a more polynomial likely

behaviour. We can notice how a great number of constraints affect the scalability of the solutions of the problem.

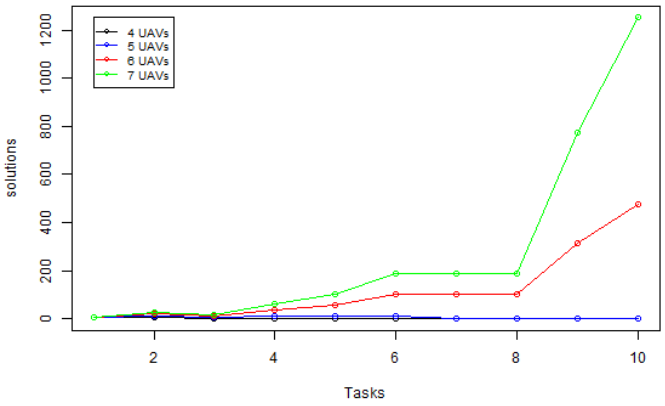


Fig. 7: Number of solutions for missions with 1 to 10 tasks, where each task collides in time with the two previous, for groups of 4 to 7 UAVs.

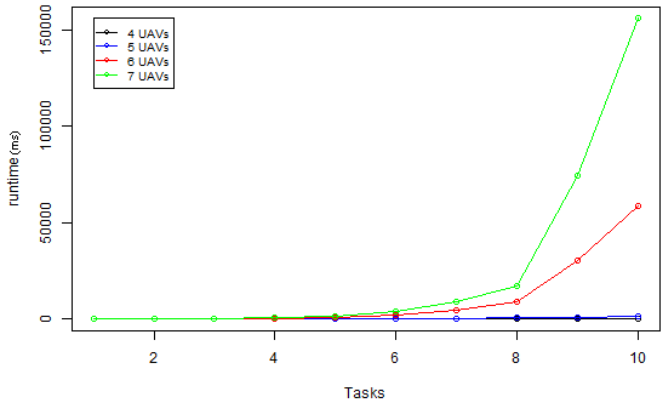


Fig. 8: Runtime for missions with 1 to 10 tasks, where each task collides in time with the two previous, for groups of 4 to 7 UAVs.

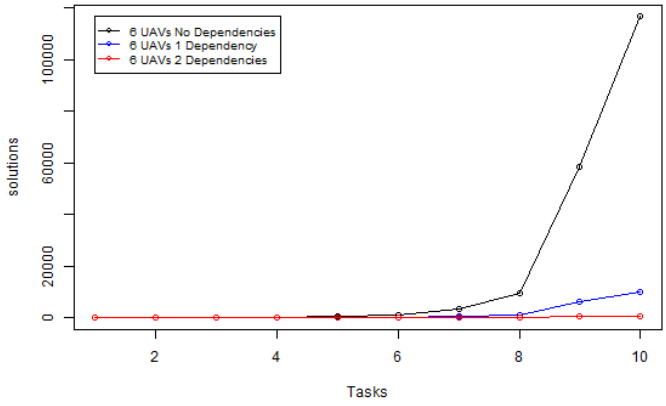


Fig. 9: Number of solutions for missions with 1 to 10 tasks for a group of 6 UAVs, with no dependencies, one dependency with the previous task or dependencies with the two previous tasks between them.

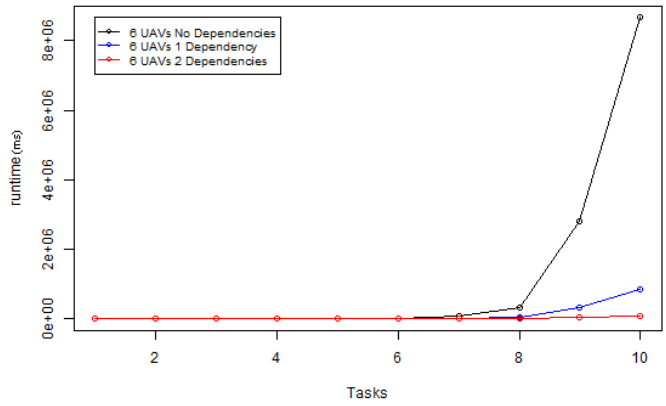


Fig. 10: Runtime for missions with 1 to 10 tasks for a group of 6 UAVs, with no dependencies, one dependency with the previous task or dependencies with the two previous tasks between them.

Finally, in Figures 9 and 10 we can see for a group of 6 UAVs, a comparison of the results obtained according to the number of existing dependencies explained in the three previous experiments. We can see how the temporal constraints highly affect the space of solutions of the problem, but also the runtime necessary to find this new space of solutions.

VII. Conclusions and Discussion

The paper presents a model for UAV Mission Planning based on Temporal Constraint Satisfaction Problems, and a scalability analysis of this problem as the number of variables increases. The presented approach defines missions as a set of tasks to be performed by several UAVs with some capabilities. The problem is modelled using: (1) temporal constraints to assure that each UAV only performs one task at a time; (2) logical constraints such as the maximum and minimum altitude reachable or restricted zone permissions, and (3) resource constraints, such as the sensors and equipment needed or the fuel consumption.

From the obtained results, we have observed that the runtime necessary to search the entire space of solutions by BT search is exponential as reported in literature. However, as the number of constraints increases (in this case the dependency constraints making tasks collide in time), the runtime decreases highly, but this scalability still resembles exponential. On the other hand, the number of solutions resembles exponential, but as the number of dependency constraints increases, the scalability loses its exponential behaviour and resembles more

polynomial. This is due to the power of a dependency temporal constraint, which highly reduces the search space of solutions.

Although the runtime needed for exploring the space of solutions is exponential, we have seen that when there are too many constraints, as the number of tasks increase, there is a point where the resources of the available UAVs needed to supply all the tasks of the mission begin to decrease. In this situation, the number of solutions begin to decrease despite the increase of possible assignments due to a higher number of tasks.

As future lines of work, this model needs to be compared against other optimization algorithms (such as Branch and Bound, Genetic Algorithms or Swarm algorithms, among others) to find feasible solutions without the necessity to explore the whole search space, which is not possible in a real scenario. Using these new algorithms, new heuristics to reduce the complexity of the problem and adapting our current model, we expect to be able to simulate problems near to real scenarios.

ACKNOWLEDGMENTS

This work is supported by the Spanish Ministry of Science and Education under Project Code TIN2010-19872 and Savier Project (Airbus Defence & Space, FUAM-076915). The authors would like to acknowledge the support obtained from Airbus Defence & Space, specially from Savier Open Innovation project members: José Insenser, César Castro and Gemma Blasco.

REFERENCES

[1] C. Schumacher, P. Chandler, M. Pachter, and L. Pachter, "Uav task assignment with timing constraints via mixed-integer linear programming," DTIC Document, Tech. Rep., 2004.

[2] W.-C. Chiang and R. A. Russell, "Simulated annealing metaheuristics for the vehicle routing problem with time windows," *Annals of Operations Research*, vol. 63, no. 1, pp. 3–27, 1996.

[3] S. Leary, M. Deittert, and J. Bookless, "Constrained UAV mission planning: A comparison of approaches," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, November 2011, pp. 2002–2009.

[4] C. Guettier, B. Allo, V. Legendre, J.-C. Poncet, and N. Strady-Lecubin, "Constraint model-based planning and scheduling with multiple resources and complex collaboration schema." in *AIPS*, 2002, pp. 284–292.

[5] C. Schulte, G. Tack, and M. Z. Lagerkvist, "Modeling and Programming with Gecode," 2010. [Online]. Available: http://www.gecode.org/

[6] P. Stuckey and M. Wallace, "Opturion-CPX," 2011-2014. [Online]. Available: http://www.opturion.com/

[7] N. Jussien, G. Rochart, X. Lorca *et al.*, "Choco: an open source Java constraint programming library," in *CPAIOR'08 Workshop on Open-Source Software for Integer and Contraint Programming (OSSICP'08)*, 2008, pp. 1–10. [Online]. Available: http://www.emn.fr/z-info/choco-solver/

[8] L. De Raedt, T. Guns, and S. Nijssen, "Constraint programming for data mining and machine learning," in *AAAI*, 2010, pp. 1671–1675.

[9] M. Mann, S. Will, and R. Backofen, "CPSP-tools - exact and complete algorithms for high-throughput 3D lattice protein studies," *BMC Bioinformatics*, vol. 9, p. 230, 2008.

[10] P. Laroche, F. Charpillet, and R. Schott, "Mobile robotics planning using abstract markov decision processes," in *Tools with Artificial Intelligence, 1999. Proceedings. 11th IEEE International Conference on*, 1999, pp. 299–306.

[11] A. O. R. R. Daniel Diaz, Amedeo Cesta and M. D. R-Moreno, "Efficient Energy Management for Autonomous Control in Rover Missions," *IEEE Computational Intelligence Magazine*, vol. 8, no. 4, pp. 12–24, 2013, special Issue on Computational Intelligence for Space Systems and Operations.

[12] U. Kuter, E. Sirin, B. Parsia, D. Nau, and J. Hendlerb, "Information gathering during planning for web service composition," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 3, no. 2-3, pp. 183 – 205, 2005, selcted Papers from the International Semantic Web Conference, 2004 ISWC, 2004 3rd. International Semantic Web Conference, 2004.

[13] D. Camacho, J. M. Molina, and D. Borrajo, "A multiagent approach for electronic travel planning," in *Proceedings of the Second International Bi-Conference Workshop on Agent-Oriented Information Systems (AOIS-2000)*. Austin, TX (USA): AAAI, 2000.

[14] D. Camacho, R. Aler, D. Borrajo, and J. M. Molina, "Multi-agent plan based information gathering," *Appl. Intell.*, vol. 25, no. 1, pp. 59–71, 2006.

[15] N. M. Gregory, G. A. Dorais, C. Fry, R. Levinson, and C. Plaunt, "IDEA: Planning at the core of autonomous reactive agents," in *in Proceedings of the 3rd International NASA Workshop on Planning and Scheduling for Space*. Sixth International Conference on AI Planning and Scheduling, April 2002.

[16] D. Camacho, C. Hernandez, and J. M. Molina, "Information classification using fuzzy knowledge based agents," in *Proceedings of the IEEE Systems, Man, and Cybernectics Conference (SMC-2001)*. USA: IEEE, 2001.

[17] D. Camacho, F. Fernandez, and M. A. Rodelgo, "Roboskeleton: An architecture for coordinating robot soccer agents," *Eng. Appl. of AI*, vol. 19, no. 2, pp. 179–188, 2006.

[18] G. Vachtsevanos, L. Tang, G. Drozeski, and L. Gutierrez, "From mission planning to flight control of unmanned aerial vehicles: Strategies and implementation tools," *Annual Reviews in Control*, vol. 29, no. 1, pp. 101 – 115, 2005.

[19] P. Doherty, J. Kvarnström, and F. Heintz, "A temporal logic-based planning and execution monitoring framework for Unmanned Aircraft Systems," *Autonomous Agents and Multi-Agent Systems*, vol. 19, no. 3, pp. 332–377, December 2009.

[20] P. Fabiani, V. Fuertes, A. Piquereau, R. Mampey, and F. Teichteil-Konigsbuch, "Autonomous flight and navigation of {VTOL} UAVs: from autonomy demonstrations to out-of-sight flights," *Aerospace Science and Technology*, vol. 11, no. 2-3, pp. 183 – 193, 2007.

[21] F. Teichteil-Konigsbuch and P. Fabiani, "A multi-thread decisional architecture for real-time planning under uncertainty," in *3rd Workshop on Planning and Plan Execution for Real-World Systems, Providence, RI*, 2007.

[22] F. Adolf and F. Andert, *Onboard mission management for a VTOL UAV using sequence and supervisory control*. InTech, October 2010, ch. 19, pp. 301–316.

[23] F. Kendoul, "Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems," *J. Field Robot.*, vol. 29, no. 2, pp. 315–378, March/April 2012.

[24] B. Bethke, M. Valenti, and J. P. How, "UAV Task Assignment," *IEEE Robotics and Automation Magazine*, vol. 15, no. 1, pp. 39–44, March 2008.

[25] M. Alighanbari, "Task Assignment Algorithms for Teams of UAVs in Dynamic Environments," Master's thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, Cambridge MA, June 2004.

[26] J. Kvarnström and P. Doherty, "Automated planning for collaborative UAV systems," in *Control Automation Robotics & Vision (ICARCV)*. IEEE, December 2010, pp. 1078–1085.

[27] R. Barták, "Constraint programming: In pursuit of the holy grail," in *In Proceedings of the Week of Doctoral Students (WDS99 -invited lecture*, 1999, pp. 555–564.

[28] A. Gonzalez-Pardo and D. Camacho, "A new CSP graph-based representation for ant colony optimization," in *2013 IEEE Conference on Evolutionary Computation (CEC 2013)*, vol. 1, 2013, pp. 689–696.

[29] A. Gonzalez-Pardo, F. Palero, and D. Camacho, "An empirical study on Collective Intelligence algorithms for Video Games problem-solving," in *The journal Computing and Informatics, to appear*, 2014, pp. 1–20.

[30] ——, "Micro and Macro Lemmings simulations based on ants colonies," in *EvoGAMES- Bio-inspired Algorithms in Games*, Granada, Spain, April 2014, pp. 1–12.

[31] C. Bessière, P. Meseguer, E. Freuder, and J. Larrosa, "On forward checking for non-binary constraint satisfaction," in *Principles and Practice of Constraint Programming CP99*, ser. Lecture Notes in Computer Science, J. Jaffar, Ed. Springer Berlin Heidelberg, 1999, vol. 1713, pp. 88–102.

[32] C. Bessière, "Constraint propagation," *Handbook of constraint programming*, pp. 29–83, 2006.

[33] P. Van Beek, "Backtracking search algorithms," *Handbook of constraint programming*, pp. 85–134, 2006.

[34] E. Schwalb and L. Vila, "Temporal constraints: A survey," *Constraints*, vol. 3, no. 2-3, pp. 129–149, 1998.

[35] M. Mouhoub, "Solving temporal constraints in real time and in a dynamic environment," AAAI, Tech. Rep. WS-02-17, 2002.

[36] ——, "Reasoning with numeric and symbolic time information," *Artif. Intell. Rev.*, vol. 21, no. 1, pp. 25–56, 2004.