EVOLVING AN ECOLOGY OF MATHEMATICAL EXPRESSIONS WITH GRAMMATICAL EVOLUTION

Manuel Alfonseca[1] − Francisco José Soler Gil[2]

[1] Escuela Politécnica Superior, Universidad Autónoma de Madrid
[2] Universidad de Sevilla & Technische Universität Dortmund

This paper describes the use of grammatical evolution to obtain an ecology of artificial beings associated with mathematical functions, whose fitness is also defined mathematically. The system allows “parasite” species and “parasites of parasites” to develop, and supports the simultaneous evolution of several ecological niches. The use of standard measurements makes it possible to explore the influence of the number of niches or the presence of parasites on “biological” diversity and similar functions. Our results suggest that some of the features of biological evolution depend more on the genetic substrate and natural selection than on the actual phenotypic expression of that substrate.

Keywords: Ecology simulation; Grammatical evolution; Mathematical functions; Ecological diversity

1. Introduction

Ecological simulation has a long history. Ever since Vito Volterra developed his famous predator-prey equations (Volterra, 1931) continuous simulation has been used to represent artificial ecological systems (Alfonseca et al, 1998). Discrete simulation has also been used frequently, using such tools as cellular automata and Lindenmayer systems (Alfonseca et al, 2003). Agent-based artificial life ecosystems are relatively old (Conrad and Pattee, 1970) and have fused with artificial life research since the end of the 1980s (see Dorin et al, 2008, for a relatively recent survey of the field). Typical recent simulations in this field tend to define predator-prey systems and complicate the agents by embodying them with fuzzy cognitive maps and similar constructs (Gras et al, 2009).

In biological evolution, a genetic substrate, embodied in nucleic acids, is subject to a certain number of random actions (mutation, recombination, etc.). The different genetic compositions are not selected directly. They are translated into phenotypes whose mutual interaction gives rise to natural selection. Our hypothesis is that many of the features of biological evolution depend more on the genetic substrate and the mechanism of natural selection than on the actual phenotypic expression of that substrate.

This paper describes our experiments to build an evolving ecology of artificial beings which compete for a limited resource environment. The underlying genetic structure is not too dissimilar to that of biological beings (a series of codons, represented as integers), but its phenotypic expression is completely different. In our artificial ecologies, the genomes are subject to genetic algorithms similar to those in biology. Grammatical evolution (GE) is then used to generate, from the genetic substrate, phenotypic counterparts completely different from living beings (a set of simple mathematical expressions). Natural selection is then applied to these phenotypes, after computing mathematically the fitness of the different individuals.

Some of the typical features of biological evolution have been reproduced successfully in this simplified environment. Other features we have found could provide new ideas about biological evolution.

Grammatical evolution, a standard technique in genetic programming (see O'Neill and Ryan, 2003, Dempsey et al, 2009, Byrne et al, 2010), suggested itself as the proper method, since it separates genomes from phenotypes and improves the *closure problem* (the need to eliminate individuals with invalid phenotypes), by protecting phenotypes against syntactic errors. Extensions to grammatical evolution, such as attribute grammatical evolution or Christiansen grammatical evolution (de la Cruz et al, 2005, Ortega et al, 2007) can also protect from semantic errors. We did not need to use those extensions, because our individuals are protected from semantic errors in a different way (see below).

Our agents are very simple, as they only embody a mathematical expression, which is executed to compare their respective fitness. Besides normal individuals, we have also introduced a second kind of agents, the parasites, whose phenotype function invokes the phenotype function of a different individual (and thus copies its fitness). The environment is also very simple: agents do not have a spatial location, although they can belong to one of several ecological niches, which evolve simultaneously, but independently. In biology, an ecological niche is a section of a population that has its own way of living and evolves relatively independently from those in other niches. We represent niches by applying different fitness functions to those individuals belonging to each niche.

Both the expressions and the grammatical evolution environment are written in the APL2 language (Alfonseca and Selby, 1989), which has been selected as the language of choice for the following reasons:

- APL2 is a very powerful language, especially for the generation of expressions, with a large number of primitive functions and operators available.
- The APL2 expression grammar is very simple and can be implemented with just four non-terminal symbols, which makes the grammatical evolution process simpler.
- APL2 instructions can be protected to prevent semantic and execution errors giving rise to program failures. In this way, we can rest assured that all the programs in the benchmark will execute (although their results may not be a good answer to the assignment). The grammatical evolution technique also becomes simpler thanks to this feature, because it is not necessary to include any semantic information.
- Being an interpretive language, APL2 makes it possible to create programming functions at execution time, thus providing the feasibility of computing fitness during the execution of the genetic algorithm. With a compiling language such as C, this would be very difficult.

This paper is divided in the following way: section 2 describes our procedure (grammatical evolution and the generation of mathematical expression phenotypes from a genome). Section 3 describes our experiments, and explains three of them in more detail. Section 4 shows the results of those three experiments, followed by a global analysis of the results of all the 200 experiments we have performed. Finally, section 5 discusses and summarizes our conclusions and lists our future work objectives.

2. Grammatical Evolution (GE)

GE is an Evolutionary Automatic Programming (EAP) algorithm based on strings, independent of the language used. Genotypes are represented by strings of integers (each of which is named codon) and the context-free grammar of the target programming language is used to deterministically map each genotype into a syntactically correct phenotype (a program). In this way, GE avoids one of the main difficulties in EAP: the results of genetic operators are guaranteed to be syntactically correct, while allowing the inclusion of multiple types.

The following scheme shows the way in which GE combines traditional genetic algorithms with genotype-to-phenotype mapping.

1) An initial population of N genomes is generated at random. A genome is a vector of n integers in the [0-255] interval. The role of each element in the genome depends on its position and is redundant (several different integers in the same position give rise to the same phenotype).
   In our experiments, the value of N is a parameter which can be set for each experiment. The value of n is random for each genome, in the [50-199] interval. We have also introduced the concept of “niche,” which makes it possible to split the population in several sub-populations, each using a different fitness function. The first element in each genome defines the ecological niche the individual belongs to.
2) The phenotypes associated to all the members in the initial population are generated, using a grammar. In our experiments, each genome is assigned a unique function number nnn in the interval [000-N). A phenotype is an APL2 function of the following form:

```
[0] Z←Fnnn X
[1] Z←(ρX)ρ0
[2] →(5<ρ⎕LC)/0
[3] '' ⎕EA 'Z←APL2_expression'
```

Only the APL2 expression in line 3 is generated from the genome. The remainder of the functions is the same for all.

- Line [0] defines a monadic function with explicit result, called Fnnn.
- Line [1] assigns to the function result a vector of zeros.
- Line [2] stops the execution of the function if function call depth is greater than 5 (this eliminates infinite recursion).
- Line [3] executes the expression generated from the genome and, if no error is detected, returns its value as the result of the function. Otherwise, a result of all zeros is returned (this is what line [1] is for).

Parasite functions can be generated by this algorithm. Parasite function expressions have the following form:

```
(Fmmm X)
```

In APL2, this expression invokes monadic function *Fmmm* (where mmm represents any three digit number) with right argument *X*. In our experiments, the value of *X* is a vector of integers from 1 to

10. The expression is written enclosed in parenthesis to isolate it from other possible expressions around it (this is a future objective). A parasite function expression thus calls a different function; for instance, function 437 may call `F521`, where mmm=521 is the function number associated to a different individual. Therefore, the parasite (individual 437) is copying the fitness value of individual 521. This makes it also possible for parasites of parasites to appear (i.e. individuals whose phenotype function invokes another function, which on its side invokes a third function).

3) The genotype population is sorted according to fitness (computed from the phenotypes). In our experiments, the fitness of a function is defined by a mathematical expression, which can be different for different ecological niches. For instance, a fitness function could be:

```
((+/|2-/2-/2-/2-/Z)+÷|+/2-/2-/2-/Z)×(.2 5)[50<ρX]
```

where Z represents the result of applying the function associated to one individual to the input values (in our experiments, all the integers from 1 to 10) and X is the genome of the individual. This fitness function selects for those mathematical expressions whose fourth difference is minimal, their third difference is maximal, (i.e. polynomials of degree 3) and whose genome is shorter than 50 integers. In APL2, `2-/`applied to a vector computes its difference (i.e. the result is a vector with one element less, whose elements are the differences between two consecutive elements in the original vector). `+/|` adds the absolute values of all the elements of the final difference vector. The fitness value is multiplied by 0.2 if the length of the genome is no longer than 50; otherwise, it is multiplied by 5, penalizing long genomes (the optimal fitness value is 0). The following expression represents this fitness function in standard mathematical notation:

$$[(\sum |\Delta^4 Z|) + (\frac{1}{|\sum \Delta^3 Z|})] \times n$$

where n=0.2 if the genome of the individual is no longer than 50 elements, and 5 otherwise.

4) The individuals in the population are ordered by their fitness. In our experiments, this is done independently for all ecological niches, so that evolution takes place independently in each niche. All those individuals whose fitness values are greater than 1000 are eliminated, together with their associated phenotype functions and all those parasites that use them.

5) Create the next generation from a mating pool. In our experiments, the mating-pool is chosen from the 100 best fitted individuals in the population (or those that remain, if they are less than 100). Four different genetic operations are applied to the offspring:

- Single point recombination of parent genomes. This operation is performed always.
- Mutation (random change of a component of the genome). This operation is performed after recombination has taken place, with 100% probability when the two parents are identical and 10% probability otherwise, to compensate the fact that recombination has no effect in that case. This is not the standard mutation procedure, but it has been used before in genetic programming (Ortega et al, 2003, Byrne et al, 2010). The first element of a genome can also mutate, which means that the offspring may belong to a different niche than their parents. This makes niche colonization after extinction possible.

- Extension: with a certain probability, a part of the genome of one parent is added at the end of the offspring genome.
- Shortening: with a certain probability, one component of the offspring genome is deleted randomly.

6) The offspring genomes are added to the population. In our experiments, if the total number of individuals exceeds N, the worst genomes in the previous population are eliminated (together with their phenotypes and parasites) until the number is N or less. The offspring genomes are associated with phenotype numbers that are of have become free after this operation.
7) The phenotypes associated to all the new members of the population are generated, using the same grammar.
8) Go to step 3.

The following grammar is used to generate a phenotype from a genotype:

```
E ::= U | U | ... | U | (FNNN X)              (right part U is repeated 30 times)
U ::= O | oO | OoO                            (non-parasite expression)
O ::= N | X | (U)                             (operands)
o ::= + | - | × | * | ÷ | ○ | ⌈ | ⌊ | ⍟ | ! | |     (operators)
N ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9        (digits)
```

where F, X, the digits and operators {+ , - ,× , * , ÷ , ○ , ⌈ , ⌊ , ⍟ , ! , |} are the terminal symbols of the grammar, while {*E* , *U* , *O* , *o* , *N* } are the non-terminal symbols, or variables, i.e. intermediate symbols that will transform into other symbols using one of the indicated rules. Table 1 shows the functions the APL2 operators compute.

Table 1: APL2 operators generated by the grammar

Operator	Monadic	Dyadic
+	Identity	Addition
-	Sign change	Subtraction
×	Sign function	Multiplication
*	Exponential	Power
÷	Inverse	Division
○	Pi times	Circular functs.
⌈	Higher integer	Maximum
⌊	Lower integer	Minimum
⍟	Natural log	Base log
!	Factorial	Combinatorial
|	Absolute value	Residue

A phenotype is generated from a genotype in the following way:

1) Variable V is initialized with the axiom of the grammar, ‘E’
2) If V does not contain a non-terminal symbol, the process has finished and the value of variable V is the phenotype expression. If step 2 has been executed 500 times, the process finishes and returns an empty expression. Otherwise:
    a. Let N be the first non-terminal symbol in V.
    b. Let K be the number of rules in the grammar whose left part is N. If K=1, the only available right part replaces the first appearance of N in V and step 2 is repeated. Otherwise:
    c. Let G be the next element of the genome under translation. If all the elements of the genome have been used, the first one is used again (genomes are circular).
    d. The first appearance of N in V is replaced by the nth right part of the rule whose left part is N (numbered in zero origin), where n=mod(G,K).
    e. Repeat step 2.

Example: let the genome be [120, 86, 37, 47, 127]. In step 1, we start with V=’E’.

1. The first non-terminal symbol in V is E. The number of right parts of the rule with left part E is K=31. The next element in the genome is G=120. Therefore n=mod(120,31)=27. The 27th right part (in zero origin) for the rule with left part E is U. We replace E by U in V. After this step, V=’U’.
2. The first non-terminal symbol in V is U. The number of right parts of the rule with left part U is K=3. The next element in the genome is G=86. Therefore n=mod(86,3)=2. The 2nd right part (in zero origin) for the rule with left part U is OoO. We replace U by OoO in V. After this step, V=’OoO’.
3. The first non-terminal symbol in V is O. The number of right parts of the rule with left part O is K=3. The next element in the genome is G=37. Therefore n=mod(37,3)=1. The 1st right part (in zero origin) for the rule with left part O is X. We replace O by X in V. After this step, V=’XoO’.
4. The first non-terminal symbol in V is o. The number of right parts of the rule with left part o is K=11. The next element in the genome is G=47. Therefore n=mod(47,11)=3. The 3rd right part (in zero origin) for the rule with left part o is *. We replace o by * in V. After this step, V=’X*O’.
5. The first non-terminal symbol in V is O. The number of right parts of the rule with left part O is K=3. The next element in the genome is G=127. Therefore n=mod(127,3)=1. The 1st right part (in zero origin) for the rule with left part O is X. We replace O by X in V. After this step, V=’X*X’.
6. Now V does not contain any non-terminal symbol, therefore the generation is completed and the result is expression ‘X*X’, i.e. X to the X power. The APL2 function generated is

```
[0] Z←Fnnn X
[1] Z←(ρX)ρ0
[2] →(5<ρ⎕LC)/0
[3] '' ⎕EA 'Z←X*X'
```

A parasite function is generated with a 1/31 probability, as there is a single right part for axiom E which generates a parasite, against 30 that give rise to ordinary arithmetical expressions.

3. Simulation experiments

In our experiments, we first define the following parameters:

- The initial and maximum sizes of the population.
- The number of ecological niches.
- Interbreeding between different niches allowed or not (applicable only if more than one niche).
- The set of values used as arguments for the phenotype functions. In all our experiments this was a vector of integers from 1 to 10.
- The fitness function(s).
- The random seed.

Each of the experiments we performed were repeated 10 to 30 times, with different values for the random seed and the interbreeding percentage, so as to have statistically meaningful results. Although in the next section we give the global results for all the 200 experiments, we will describe three of them in detail, to signal interesting effects we have detected.

1. In all our experiments with a single niche, the fitness function was

   ```
   ((+/|2-/2-/2-/2-/Z)+÷|+/2-/2-/2-/Z)×(20⌈ρX)÷100
   ```

   which selects for third degree polynomials (the smallest possible fourth difference) with a large coefficient for the third degree term (the largest possible third difference), and genomes of about 20 elements.

   In our first experiment described in more detail, which belongs to this type, we chose an initial/maximum population of 500/1000 individuals; the random seed chosen was 16807.

2. For the experiments with two niches, the fitness functions were:

   ```
   ((1E¯6+|+/2-/(2-/Z)÷1↓Z)+÷|+/2-/Z)×(.2 5)[50<ρX]
   ((+/|2-/2-/2-/2-/Z)+÷|+/2-/2-/2-/Z)×(.2 5)[50<ρX]
   ```

   The first function selects for exponential functions; the second for third degree polynomials. In both cases, genomes of less than 50 elements are positively selected (to prevent runaway genome length).

   In our second experiment described in more detail, which belongs to this type, the parameters were the following: initial/maximum population of 1000/2000 individuals; interbreeding between niches was fully allowed; the random seed was 282475249.

3. For the experiments with three niches, the fitness functions were:

   ```
   ((1E¯6+|+/2-/(2-/Z)÷1↓Z)+÷|+/2-/Z)×(.2 5)[50<ρX]
   ((+/|2-/2-/2-/2-/Z)+÷|+/2-/2-/2-/Z)×(.2 5)[50<ρX]
   ((+/|2-/2-/2-/2-/2-/Z)+÷|+/2-/2-/2-/2-/Z)×(.2 5)[50<ρX]
   ```

   The first function selects for exponential functions; the second for third degree polynomials; the third for fourth degree polynomials, in all three cases with the largest possible absolute value. Genomes of less than 50 elements are positively selected to prevent runaway genome length.

For our third detailed experiment, which belongs to this type, we chose an initial/maximum population of 2000 individuals; no interbreeding between niches was allowed; the random seed chosen was 16807.

To select our fitness functions, we have tried to combine mathematical expressions which can be considered “near” and “far”, measured by their genetic distance. For example, the distance between a third and a fourth degree polynomial is smaller than the distance from each of them to an exponential function. To see it, notice that function ‘x*3’ can be converted to ‘x*4’ by a single mutation (and vice versa) while two mutations are needed to convert ‘x*3’ to ‘3*x’. In future experiments, we intend to measure the relative ease with which the offspring generated in our experiments can migrate from one niche to another, depending on the genetic distance of their fitness functions.

Biological populations are almost never genetically identical, they embody a certain degree of variation, even when they belong to a single species. Among different ways to measure biodiversity, the Shannon diversity index (Shannon, 1948) is frequently used. This index is defined by the following formula:

$$-\sum_{i=1}^{n} p_i \log\ p_i$$

where n is the number of different species and p_i is the frequency of species number i (the number of individuals belonging to that species divided by the total number of individuals).

To study the evolution of diversity in our simulation experiments by means of Shannon’s diversity index, we group the individuals in “species.” Two individuals belong to the same “species” when their phenotypes are identical (even though their genotypes may not be, due to the redundancy of the genetic code). Phenotypes are considered identical when the mathematical expressions in their phenotype functions are identical. Expressions that always give rise to the same values, but are not identical, are considered to belong to different species. For instance, $(\ 2\times X\ ) * 3$ and $8\times X * 3$ are different species, even though their results (and therefore their fitness value) are always the same.

4. Results

Table 2 shows some of the dominant functions (those which reached the maximum fitness) during the evolution of the ecology simulated in our first detailed experiment.

Table 2: Evolution of the dominant function in the first experiment

Dominant function	Generation	Fitness
$(2X)^3$	100	0.00083
(F023 X)	300	0.00059

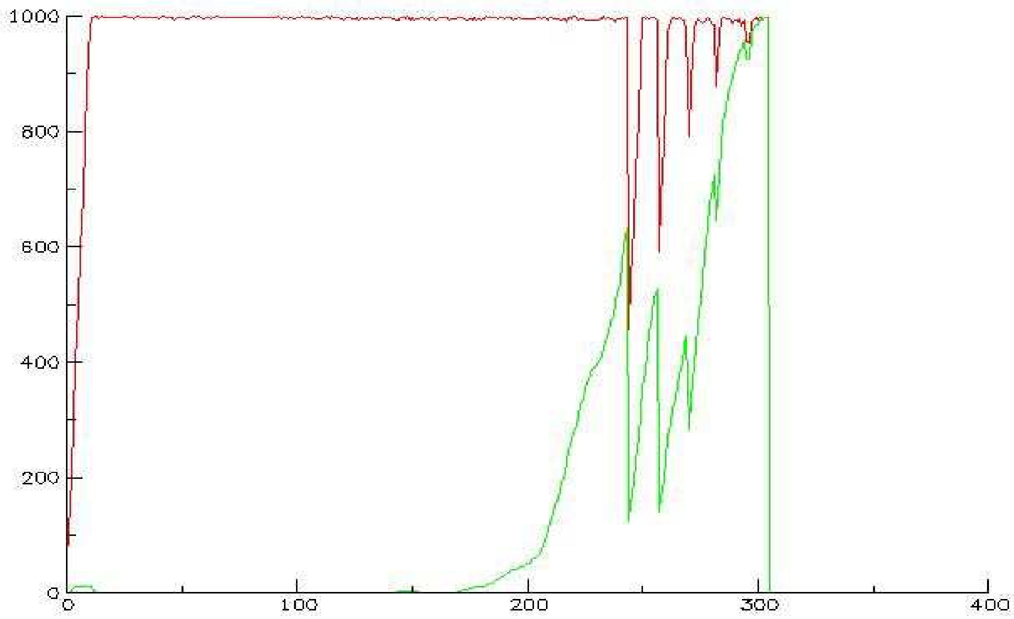

Figure 1: Results of the first experiment: red, total population size; green, number of parasites.

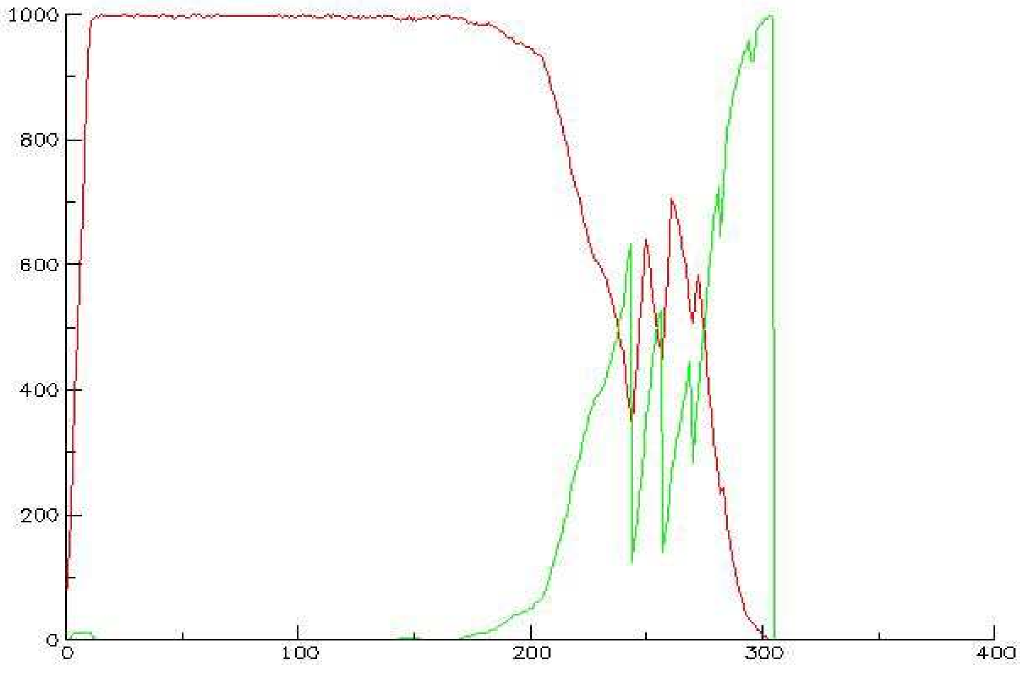

Figure 2: Results of the first experiment: red, normal population size; green, number of parasites.

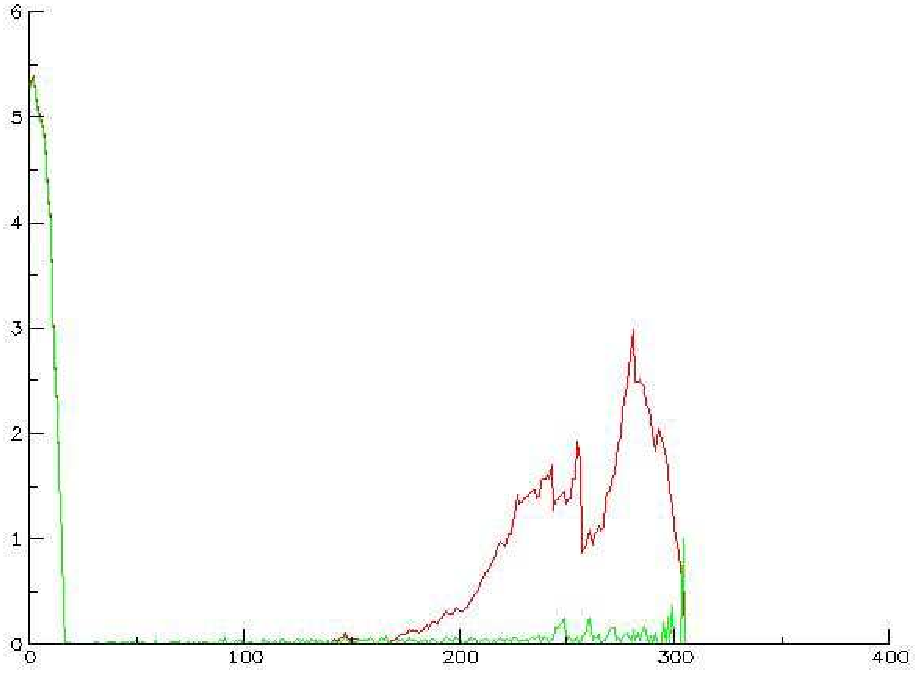

Figure 3: Shannon's diversity index for the first experiment: red, total population; green: normal population.

This ecology endured for 305 generations, then disappeared (the size of its population became zero). Figure 1 shows the total size of the population as a function of the generation number, as well as the total number of parasite individuals. Figure 2 displays the evolution of the normal individuals and the parasites. It is easy to see that, since parasites and normal individuals are competing for reproduction, but parasites also depend on their hosts to survive, the relationship between them becomes somewhat similar to that between prey and predator described by Volterra-Lotka equations. It is clear in the figures that, when parasites get into a runaway situation, they may end up destroying the whole population (as in generation 305) or stopping their own development (this happens several times in this experiment), in which case the normal population is able to recover. An interesting feature is the appearance of parasites of parasites.

Figure 3 shows the value of the Shannon diversity index in this experiment, in red for the total population, in green for normal individuals only. It can be seen that the latter diversity is always very low in this experiment, with index values usually much smaller than 1.

Table 3 shows some of the dominant functions in our second detailed experiment.

Table 3: Evolution of the dominant functions in the second experiment

Niche 1		Niche 2	
Dominant function	Generation	Dominant function	Generation
e^{30X}	100	$8X^3$	100
e^{12X+6}	600	$9\times(9X)^3$	400
e^{9X}	1100	$X(X+6)\log_4 X$	800
		$8!\times X^2(3-X)$	900
		$9\times 9!\times X^3$	1100
		$9\times 9^9\times 9!\times X^3$	1500

This ecology seemed to endure more or less permanently, so we stopped the experiment after 2000 generations. Figure 4 shows the total size of the population as a function of the generation number, as well as the total number of parasite individuals. It can be seen that there is a big parasite attack, during which the number of parasites explodes and the number of normal individuals drops, until the loss of hosts puts a stop on the parasite runaway growth. Figure 5 displays a zoom on the normal individuals and the parasites during the parasite attack (generations 400 to 600).

Figure 6 shows what happened to the two niche populations during the experiment. It can be seen that the population of the second niche totally disappeared a little before generation 500, obviously as a result of parasite runaway growth for that niche. As soon as this happened, all the parasites for the second niche also disappeared. The system then became essentially a one-niche system, and remained like that for over 200 generations. Due to a curious chance, the first niche was also affected by parasites at the same time (in all our two niche experiments, this only happened once). First niche parasites then also experimented a runaway growth which was almost put an end to the experiment (the population of the first niche came down to 5 at generation 571), but the collapse of the parasites saved the issue and those 5 individuals were able to recover, reaching again the maximum population at generation 600. Later, just before generation 800, the second niche was colonized by one mutated individual, progeny of two individuals of the first niche, and a new evolution started at that niche from a lower fitness, which later grew to a greater fitness than that reached before (see Table 3).

Figure 7 shows the value of Shannon's diversity index in this experiment. It can be seen that the diversity is larger than in the previous experiment, remaining most of the time between 1 and 2, and reaching a maximum value of 6 just when the second niche was being re-colonized.

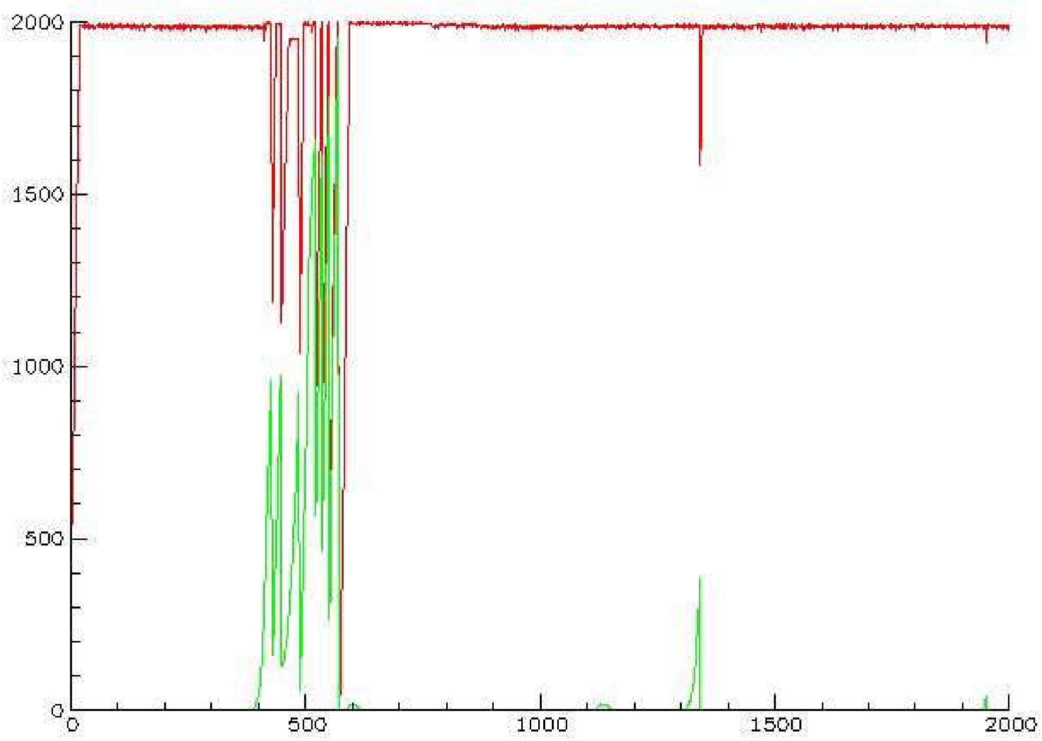

Figure 4: Results of the second experiment: red, total population size; green: number of parasites.

A similar situation took place in another experiment with three niches, full interbreeding and a population of 2000 individuals. In this case, the third niche was wiped out between generations 545 and 660. Figure 8 shows how the population was divided into the three niches.

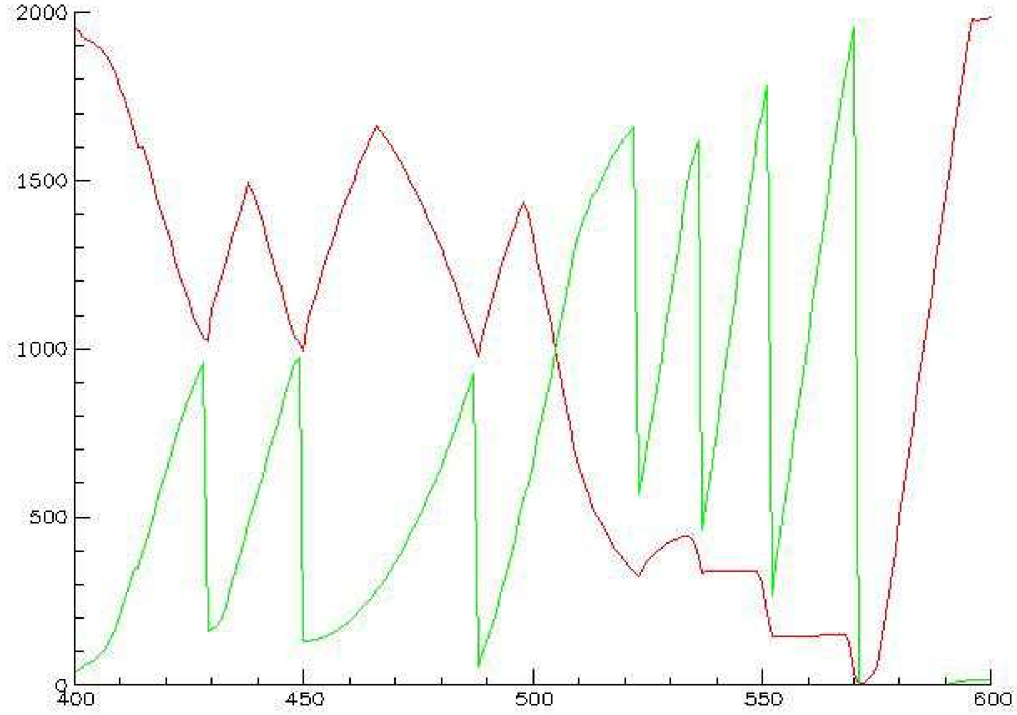

Figure 5: Zoom on the results of the second experiment: red, normal population size; green: number of parasites.

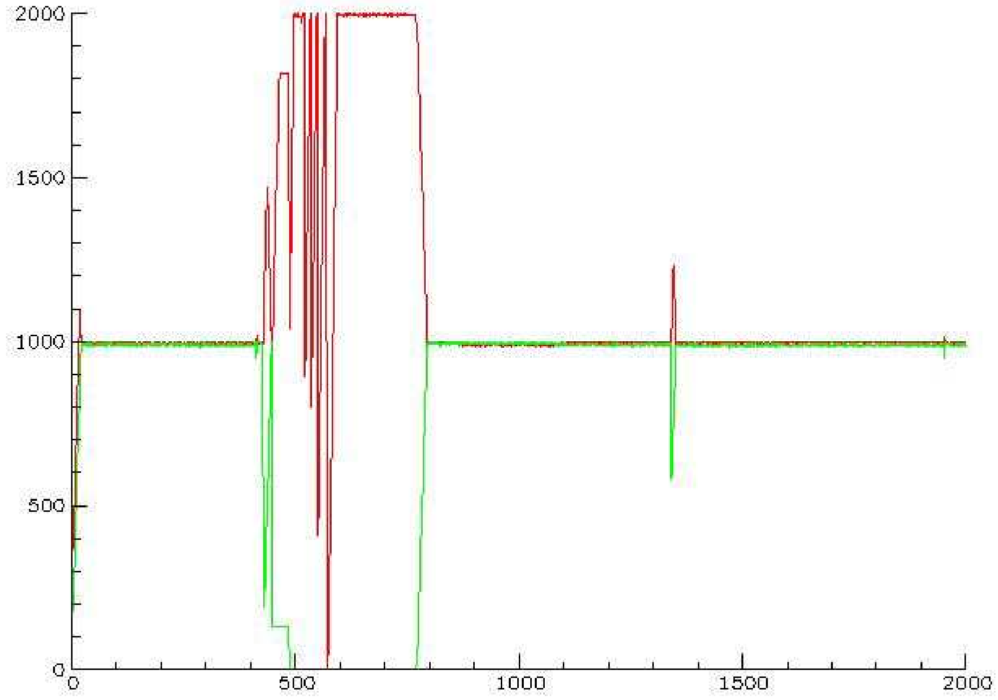

Figure 6: Population distribution among the two niches for the second experiment: red, first niche; green: second niche.

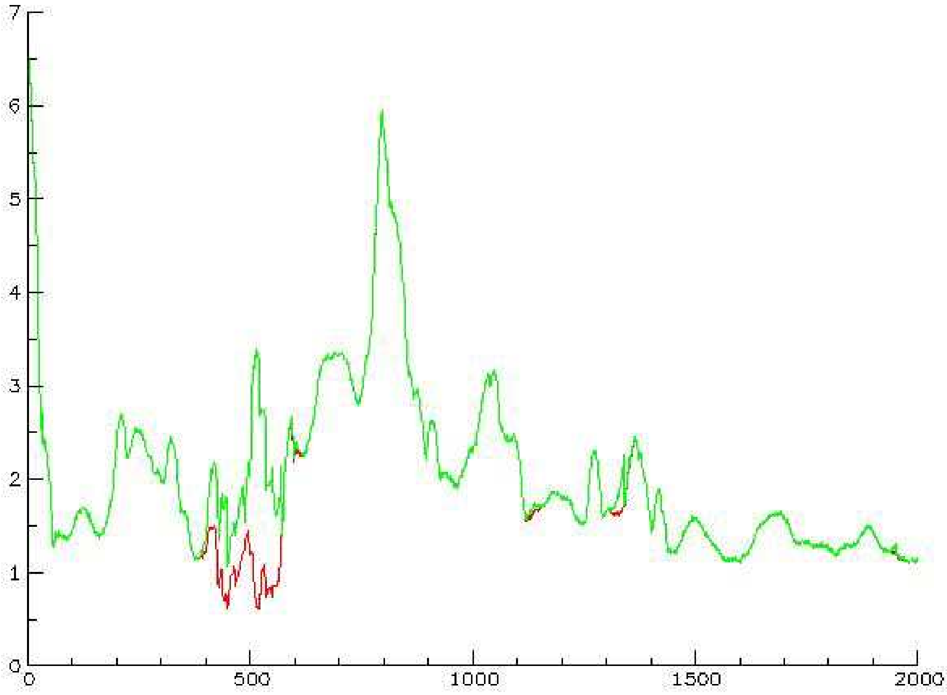

Figure 7: Shannon's diversity index for the second experiment: green, total population; red, normal population.

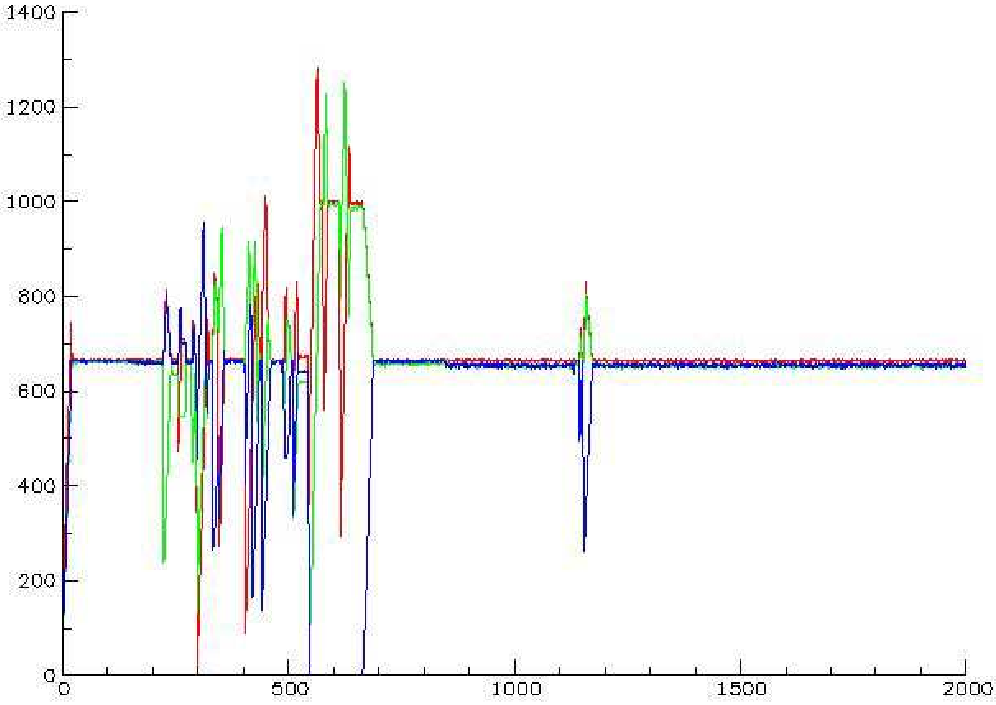

Figure 8: Population distribution among the three niches for another experiment: red, first niche; green: second niche; blue: third niche.

Table 4 shows some of the dominant functions during the evolution of our third detailed experiment. Observe that the combinatorial numbers in the last two rows for niches 2 and 3 are actually polynomials of the third degree, and in the third niche they are multiplied by X, to become a fourth degree polynomial, as expected.

In this experiment, the total number of individuals is practically constant and equal to the maximum population. There are very few parasites (69 individuals at most) during a few generations. Figure 9 represents the evolution of diversity, measured by the Shannon index. The value of diversity is also larger than in the first two experiments: it remains most of the time at a value between 2 and 3.

Table 4: Evolution of the dominant function in the third experiment

Niche 1		Niche 2		Niche 3	
Dominant function	**Generation**	**Dominant function**	**Gener.**	**Dominant function**	**Gener.**
$(3\times7!)^X$	100	$\frac{-4}{X}$	100	$\frac{5}{X}$	100
$(9\times7!)^X$	700	$X-\binom{X}{3}-3$	800	$X(X-\binom{X}{3})$	900
		$-8^{324}\binom{X}{3}$	1800	$\left(X-\binom{X}{3}\right)(36\pi7!\,X)$	1700

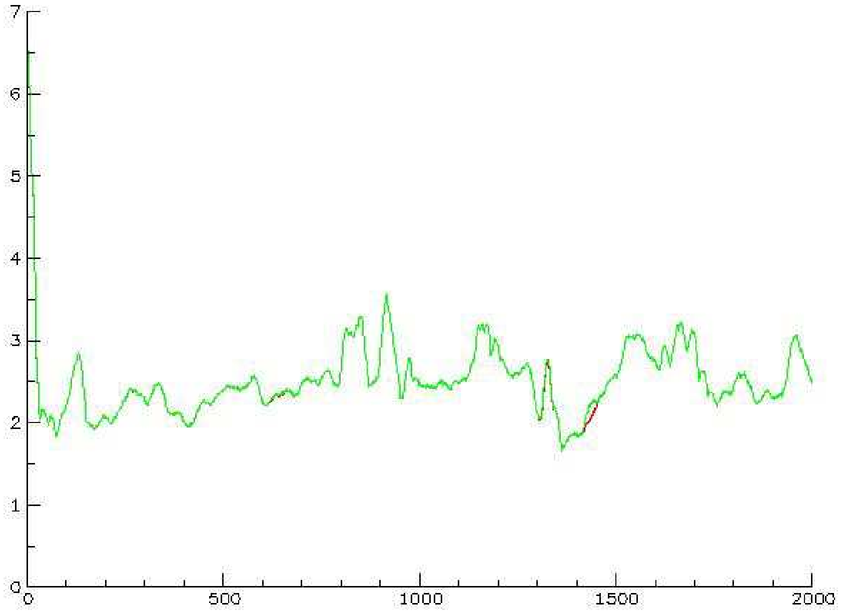

Figure 9: Shannon's diversity index for the third experiment: green, total population; red: normal population.

We have performed a total of 200 experiments, with one, two and three niches, with three different degrees of interbreeding (100%, 50% or 0%), with equal or unequal partitioning of the population. The latter is obtained by increasing the number of niches, but having some of them share the same fitness functions. For instance, in some cases we used four niches with interbreeding, two of them with the same fitness function, which gave rise to a population partition between niches of 50%, 25% and 25%. The best individuals in the two niches sharing the same fitness function were almost always identical, since interbreeding made it highly probable that the best species of one of the two niches sharing the same fitness function would spread to the other. Usually, when a mutation to some offspring makes it change niches, they will be speedily eliminated, as their fitness will be much lower than that of the dominant species in the target niche, but this will not be the case when both fitness functions are very similar, or when the niche has become empty, as in the second detailed experiment. This, and the smaller number of parasites, also explains why early extinction (as in the first experiment) never takes place in the multi-niche case.

Table 5 shows the global results we have obtained in our 200 experiments. All of them were performed with an initial population of 1000, except for those with only one niche, where the initial population was 500. Only for a single niche and a maximum population of 1000 some of the experiments (40%) ended in premature extinction. The rows marked 3* were executed with 4 niches, two of which share the same fitness function, which corresponds to 3 niches with unequal population distribution (50/25/25%). In all the measurements computed, the first 15 generations are excluded, to allow the ecology to go into a permanent regime. The last two columns show the maximum number of normal/parasite species, and the average diversity during the experiment, with and without considering parasites. In the last three columns, averages were taken for all the experiments in each family.

Table 6 details the effect when different degrees of interbreeding are allowed between niches. The reason why this was tested is because, in biological systems, individuals that occupy different niches may belong either to the same or related species, or to completely different species. In the first case, niche interbreeding would be allowed, in the second it would be forbidden. Depending on the nearness of the corresponding species, interbreeding can also be partial.

In some experiments where interbreeding is allowed, we have detected some evolutionary flow between niches 2 and 3. The best functions are first found at niche 2, then copied by niche 3, or vice versa. This happens when two individuals, one belonging to each niche, produce hybrids which copy sections of expressions from one niche to the other. This is probably made possible because the fitness functions (third and fourth degree polynomials) for those two niches are related.

Table 5: Summary of all the experiments

Number of niches	Nr. of exper.	Max. population	Extinct before gen. 2000	Max. parasites	Max. species	Aver. diversity
1	10	500	0	152	28/4.1	0.46/0.37
1	10	1000	4	939	47/16	1.08/0.55
2	30	1000	0	93	35/2.4	1.66/1.65
3	30	1000	0	57	37/2.6	2.07/2.07
3*	30	1000	0	60	41/3.2	2.25/2.24
2	30	2000	0	380	210/4.8	1.78/1.76
3	30	2000	0	209	192/3.5	2.12/2.11
3*	30	2000	0	404	218/4.7	2.30/2.27

The following behavior can be observed at table 5:

- The maximum number of parasites seems to be correlated with the number of individuals per niche, obtained by dividing the total population by the number of niches. Runaway parasite growth happens mostly for a single niche. With more niches and greater population sizes, the effect of the parasites is smaller, although one runaway (but not fatal) parasite growth occurred once for both 2 and 3 niches and a 2000 max population.
- It can be seen that the maximum number of normal species depends only on the size of the population, regardless of the number of niches. The number of parasite species, however, has a more complicated dependence.
- The occurrence of more than one niche has the following effects:
  - Ecologies are more stable and never get completely extinct (up to 2000 generations).
  - The influence of parasites is reduced (they almost never go into a runaway growth).
  - The diversity of the population increases proportionally to the number of niches. If our general hypothesis is correct, this effect was to be expected, as a greater number of niches with independent evolution must give rise to a greater diversity. This has been detected (theoretically and in practice) in real biological ecosystems.

Table 6: The effect of interbreeding

Max. population	Number of niches	Inter-breed.	Max. parasites	Max. species	Aver. diversity
1000	2	100%	134	42/3	1.62/1.61
		50%	122	39/2.7	1.78/1.78
		0%	24	24/1.6	1.57/1.57
	3	100%	103	51/2.6	2.11/2.10
		50%	37	32/3,7	2.08/2.08
		0%	30	28/1.5	2.01/2.01
	3*	100%	71	44/2.9	2.18/2.17
		50%	80	49/4.7	2.43/2.42
		0%	30	31/2	2.15/2.14
2000	2	100%	795	215/6.4	1.73/1.67
		50%	209	219/4.2	1.92/1.91
		0%	135	198/3.7	1.71/1.69
	3	100%	490	203/4.4	2.21/2.19
		50%	75	194/3.4	2.10/2.10
		0%	63	179/2.6	2.06/2.05
	3*	100%	544	227/4.6	2.41/2.38
		50%	566	223/5.8	2.32/2.26
		0%	103	204/3.8	2.16/2.16

The following behavior can be observed at table 6:

- Allowing full interbreeding between niches gives rise to a much higher number of parasites (2 to 7 times larger than disallowing it).
- The maximum number of parasites seems to be positively correlated with interbreeding.
- The maximum number of normal and parasite species tend to decrease when niche interbreeding is reduced or not allowed.
- Diversity does not depend (or depends very little) on interbreeding.
- 50% interbreeding gives numbers intermediate between both extremes, although sometimes they get near to either one of them.

5. Conclusions

In this paper we have designed a procedure that generates artificial ecologies that exhibit some of the features of natural evolution, among them the following:

- The appearance of parasite species. Parasites of parasites also happen.
- A Volterra-like relationship between parasites and hosts.
- Different numbers of niches are supported, with and without interbreeding. Average diversity increases with the number of niches.
- When a niche becomes empty, it can be colonized by the offspring of a different niche.

To do this, we have used the following ideas:

- Grammatical evolution, which separates genomes from phenotypes (this is a standard technique in genetic programming).
- Individual genotypes are represented by means of mathematical expressions. Fitness functions become simple mathematical tests on those expressions. Mathematical expressions (using lambda-calculus, rather than APL2, without grammatical evolution) have been used before in artificial life experiments (Fontana, 1991, Fontana and Buss, 1996). In principle, lambda-calculus and APL2 should be equivalent for the representation of mathematical functions.
- Simultaneous evolution of several “niches” is attained by means of changes in the genome interpretation (the first element selects the niche), and by using several fitness functions (one per niche). We believe this is an original idea.
- Parasites are represented as individuals whose phenotype function invokes the phenotype function of a different individual (and thus copies its fitness). In the way we have implemented it, this is an original idea. Previous work in the area of artificial life, such as TIERRA (Ray, 1992, Ray and Haxt, 1998), also gave rise to parasites, but based on a different approach (individuals were computer programs executing in a virtual machine, and parasites jumped to sections of those programs).

We believe our results provide support for the hypothesis that many features displayed by biological evolution depend on chance modifications of the genome plus natural selection, rather than on the particular form adopted by the phenotypes. Therefore, our experiments could help detect other features, not so easy to discover in biology, such as some of the dependences indicated in the previous section.

In the future we intend to explore the following issues:

- Partial interbreeding between different niches.
- To measure the relative ease with which the offspring generated during our experiments can migrate from one niche to another, depending on the distance of their fitness functions.
- To extend the expressions representing parasites, allowing for a call to another function to be embedded in a more complicated expression.
- To complicate the ecology by introducing predators.

References

Alfonseca, M., Selby, D., 1989. APL2 and PS/2: The language, the systems, the peripherals, APL Quote Quad (ACM SIGAPL), vol. 19:4, pp. 1–5.

Alfonseca, M., Carro, R., de Lara, J., Pulido, E., 1998. Education in Ecology at the Internet with an Object-Oriented Simulation Language, Proceedings of Eurosim 98, pp. 118-123.

Alfonseca, M., Ortega, A., Suárez, A., 2003. Cellular automata and probabilistic L systems: An example in Ecology. In Grammars and Automata for String Processing: from Mathematics and Computer Science to Biology, and Back, ed. C. Martin-Vide & V. Mitrana, Taylor & Francis, pp. 111-120.

Byrne J., McDermott J., O'Neill M., Brabazon A., 2010. An analysis of the behaviour of mutation in Grammatical Evolution. Proceedings of the 13th European Conference on Genetic Programming, EuroGP 2010, LNCS, Vol. 6021, pp. 14-25, Springer.

Conrad, M., Pattee, H.H. , 1970. Evolution Experiments with an Artificial Ecosystem. Journal of Theoretical Biology, 28: p. 393-409.

de la Cruz, M., Ortega, A., Alfonseca, M., 2005. Attribute grammar evolution, in Lecture Notes in Computer Science, vol. 3562, pp. 182–191.

Dempsey I., O'Neill M., Brabazon A., 2009. Foundations in Grammatical Evolution for Dynamic Environments. Studies in Computational Intelligence, Vol. 194, Springer.

Dorin, A., Korb, K.B., Grimm, V., 2008. Artificial-Life Ecosystems: What are they and what could they become? Artificial Life XI: Proceedings of the Eleventh International Conference on the Simulation and Synthesis of Living Systems, MIT Press.

Fontana, W., 1991. Algorithmic chemistry. Artificial Life II, SFI Studies on the Sciences of Complexity, vol. X. Langton, C.G., Taylor, C., Farmer, J.D., Rasmussen, S., eds., Addison-Wesley, pp. 159-209.

Fontana, W., Buss, L.W., 1996. The barrier of objects: From dynamical systems to bounded organizations. In “Boundaries and barriers”, Casti, J. and Karlqvist, A., eds., Addison-Wesley, pp.56-116.

Gras, R., Devaurs, D., Wozniak, A., Aspinall, A., 2009. An Individual-Based Evolving Predator-Prey Ecosystem Simulation Using a Fuzzy Cognitive Map as the Behavior Model, Artificial Life Vol. 15:4, p. 423-463.

O’Neill, M., Ryan, C., 2003. Grammatical Evolution: Evolutionary Automatic Programming in an Arbitrary Language. Kluwer, Norwell, MA.

Ortega, A., Dalhoum, A.A., Alfonseca, M., 2003. Grammatical evolution to design fractal curves with a given dimension, IBM Journal of Research and Development, Vol. 47:4, pp. 483--493.

Ortega, A., de la Cruz, M., Alfonseca, M., 2007. Christiansen grammar evolution: Grammatical evolution with semantics, IEEE Transactions on Evolutionary Computation, vol. 7:1, pp. 77–90.

Ray, T.S. Evolution, 1992. Ecology and optimization of digital organisms. Santa Fe Institute Working Paper 92-08-O42.

Ray, T.S., Haxt, J., 1998. Evolution of diffexenfiated mulfi-thxeaded digital organisms. Artificial Life VI Proceedings, ed. C.Adami et al., pp. 295-304, The MIT Press, Cambridge.

Shannon, C. E., 1948 A mathematical theory of communication. The Bell System Technical Journal, 27, 379-423 and 623-656.

Volterra, V, 1931. Leçons sur la Théorie Mathématique de la Lutte pour la Vie, Gauthier-Villars, Paris.