Capítulo 1

Experimental Analysis

In the previous chapter we have presented the EvAg model as the framework with which this thesis will analyse the viability of the P2P EC paradigm. In addition, a first insight on the *computational performance* of the approach has been provided showing that, for very demanding problems, it is able to scale following linear speedups.

Nevertheless, such results do not take into account whether the algorithm is able to converge to convenient solutions in spite of the run-time dynamics of P2P systems. Hence, we propose in this chapter the experimental analysis of the model in a simulated P2P environment so that the viability of the P2P EA can be drawn from the *algorithmic performance* of the approach. All the source code for the experiments has been published under GPL v3 and is available from our Subversion repository at `https://forja.rediris.es/svn/geneura/evogen`.

Simulations will allow the execution of controlled experiments tackling the goals exposed in Section 1.1 in which a set of representative test-cases are proposed for studying the viability of the model. In order to define the system with a good level of detail, Section 1.2 describes the decision-making process that we have followed for the experimental analysis. Taking into account such decisions, Section 1.3 presents the overall methodology that will be followed in Sections 1.4, 1.5 and 1.6 tackling the different test-cases. Finally, conclusions on the *scalability* and *fault tolerance* of the model are drawn in Section 1.7 based on the analyis of results.

1.1. Goals

As exposed at the introduction of this thesis, the main motivation behind P2P EAs is tackling those large problem instances in which, due to memory

or computational constraints, sequential approaches are unsuitable. In this sense, analysing the scalability of the EvAg model is key to determine the approach viability. Additionally, as the sizes of the problem instances increase, the number of computing nodes required to tackle the problem scales and, therefore, failures become more likely at run-time. Taking that into account, we propose an experimental analysis focusing on the following goals in order to prove the model viability:

1. Analysing the *scalability* of the model to demonstrate that the approach is suitable for tackling large problem instances in a failure-free environment.

2. Studying the *fault-tolerance* of the model under churn conditions to demonstrate that the model follows a *graceful degradation* and is able to tackle large problem instances in spite of nodes departing from the system.

Studying the scalability makes possible not only to analyse instances under examination but also predicting the model behaviour when tackling larger instances. On the other hand, fault tolerance is a key issue in a P2P EA since churn is inherent to P2P systems given that peers are prone to failures.

In order to tackle such goals, the following test-cases have been designed to assess the algorithm performance:

1. *Scalability of the model in failure-free environments against sequential approaches.* So that the EvAg model is compared against a canonical SSGA and a GGA to demonstrate that such a spatially-structured EA scales better than panmictic schemes of evolution.

2. *Influence of the population structure on the algorithm performance.* As explained in Section **??**, there is no reason preventing the EvAg model to use population structures other than newscast. Therefore, this test-case aims analysing the scalability of the model using two common topologies in fine-grained approaches: a ring lattice [GTTA05] and a Watts-Strogatz [GPT06] population structures.

3. *Fault tolerance of the model.* In this test-case, we will analyse the scalability of the model using several scenarios of churn in which the system degrades under different failure rates.

1.2. Rationale for the experimental analysis

Given the huge complexity of either P2P systems or EAs, a detailed analysis on their interactions in the P2P EC paradigm remains beyond the scope of this thesis. Hence, a certain number of decisions has to be made in order to focus the analysis on the specifications of the goals. Therefore, this Section aims to justify the simplifications and assumptions made on the model.

1.2.1. Simplifications

In order to simplify the analysis on the model performance, the following considerations were made on the design of experiments:

- The experimental analysis in this thesis will concentrate on *binary-coded GAs* [ES03] within the rest of EC paradigms such as ES [Rec94], EP [Fog94] or GP [Koz92]. The main reason is that the population sizing theory [Gol02] that we will follow to perform analysis on the scalability of the algorithm focuses on such kind of EAs. Nevertheless, taking into account that *binary-coded GAs* follow the same evolutionary scheme than other EAs such as *real-coded GAs* [ES93] or GP, conclusions should be easily extended to other paradigms.
- With the aim of establishing a worst-case analysis, evolutionary operators will not apply specific knowledge on the problem. In this context, we will use *uniform crossover*, *bit-flip mutation* as operators and *binary tournament* as decentralised selection mechanism throughout all the experiments [ES03]. Taking into account that we will use trap functions as benchmark (described in Section 1.3.1), either *uniform crossover* or *bit-flip mutation* prevent the algorithm search to form higher order BBs, thus challenging the GA's search mechanisms. On an averaged sense, such operators are fooled by traps as described by Deb and Goldberg in [DG91].
- In the same line than the previous point, individuals will be *initialised at random*. Every gene will have an uniform probability of 0,5 to be either 1 or 0, so that, on average, the randomly generated initial population is placed on local optimum attracting regions for the problem landscapes proposed in Section 1.3.1.

1.2.2. Assumptions

The experimental analysis in this chapter examines the main influencing variables in the EvAg performance. To that aim, experiments were designed

in such a way that variables under study can be analysed by assuming fixed conditions to the rest of factors. Such assumptions are detailed in the following points:

- We have used selectorecombinative versions of the algorithms (without mutation) for estimating the population sizes. Lobo and Lima state in [LL07] that the assumption of a selectorecombinative GA is commonly made in population sizing studies as the only source of diversity is then the initial population which stands for a worst case analysis. However, this thesis complements the study analysing the convergence of the approach using mutation.

- We assume that every EvAg behaves as a *virtual node* [DHJ+07] in such a way that every physical node can host more than a single EvAg . Hence, the number of *virtual nodes* hosted in a physical one can be decided at a load-balance level depending on node capacities and a heterogeneous system such as P2P can be assumed as homogeneous at a virtual level.

- Despite having assumed homogeneous conditions in the previous point, the lack of a central clock in a decentralised scheme implies an asynchronous execution. In this sense, the most commonly used methods to simulate asynchrony either in cellular automaton [SdR99] or cellular evolutionary algorithms [Tom05] are:

  - *Fixed line sweep*, the idea here is that the update on the different cells (i.e. EvAg in our case) follows a prestablished sequential order, e.g. from left to right in rings structured populations.
  - *Fixed random sweep*, it consists in a small variation on the previous method in which the update sequence is set randomly at the beginning of a run as a permutation of the different cells.
  - *New random sweep*, a new random cell permutation is generated at every cycle of a run, understanding a cycle (or time step) as the sequential update of all the cells.
  - *Uniform choice*, the next cell to be updated is randomly chosen with uniform probability from all the possible cells. This method implies that the same cell could be updated more than once during a cycle and, in turn, some others could remain without updating.

  To the choice of an adequate update policy for our approach, the *uniform choice* policy seems more appropriate to model a decentralised run

in which there is no guarantee of any sequential order in the update of the individuals. This way, we have adopted such a policy to simulate the asynchronous update of EvAgs .

In addition, it has to be considered that Dorronsoro et al. show in [DAGT04b] that the four methods do not always present statistical differences between each other in a test-suite of four different problems. The uniform choice method presents statistical differences from the rest in a single case, indicating that choosing between these methods does not have a great influence on the algorithm performance.

- Every time that an EvAg performs a fitness evaluation, it also initiates a cache exchange of the newscast protocol, i.e. $t_r = cycle$ where t_r is the parameter for the newscast updating frequency exposed in Section **??** and *cycle* is defined as the time the algorithm takes to perform n evaluations in a population of n EvAgs using *uniform choice*.

- The evolutionary algorithm will begin at $t_r = 20$ to guarantee that the newscast protocol bootstraps and converges given that it has already shown in Section **??** that the protocol takes at around 12 cycles to converge to an state of dynamic equilibrium for a network size of 1600. In this sense, we have assumed a synchronised start up of the experiments.

- The cache size (c) is the only tunable parameter in newscast and we use $c = 20$ within all the settings of the experiments. Such value takes into account Jelasity and van Steen recommendations in [JvS02] stating that the intended normal setting of newscast is $c \ll n$ and demonstrating that values from $c = 20$ prevent the spontaneous partitioning of the graph even when it becomes very large (see Section **??** for more details). In addition, we make experiments for the fine-tuning of the parameter in the appendix of this thesis showing a lack of influence of c on the EvAg performance when $c \in [0{,}005n, 0{,}15n]$, where n is the population size.

- We have assumed that the time required for communications is negligible. Such an assumption might be unrealistic for small problem instances, but it turns feasible for problem instances becoming large. As it has been shown in the analysis of the parallel performance of the model in Section **??**, communications do not inflict a penalisation on the algorithm speed-up for problems with a very demanding fitness evaluation cost.

1.3. Experimental Methodology

In order to analyse the scalability of the EvAg model, experiments are conducted on trap functions (described in Section 1.3.1). The purpose is investigating how population sizes scale with increasing problem size and difficulty. To this end, we follow the population sizing theory [Gol02] which states that there is an optimal population size for a given problem instance that can be determined under some conditions. In particular, we use the bisection method (Section 1.3.2) that determines the minimum population size N for a selectorecombinative GA. In addition, we look at the scale-up properties of the average number of evaluations to a solution (AES), which is a device independent measure of computational effort that can be used for all EAs, with or without mutation [ES03]. Finally, the study is complemented switching mutation on, so that, the algorithm convergence is analysed for the most demanding problem instances using the metrics proposed in Section 1.3.3. Such results will be statistically analysed using the non-parametric Wilcoxon test.

1.3.1. Test-suite

Experiments were conducted on deceptive, quasi-deceptive, and non-deceptive trap functions [Ack87] following Lobo and Lima's recommendations in [LL07] about choosing a test suite with known population requirements and investigating the scalability on landscapes with different characteristics. These functions represent a set of decomposable problems based on unitation and composed of (m) sub-functions in which the total fitness is additively calculated by summing the partial fitness of every sub-function. Hence, it is easy to scale the problem from small to large instances by considering a smaller or larger number of sub-functions (m). In addition, each sub-function is composed of (k) bits representing the BB size, every sub-function is then composed of 2^k combinations from which only one belongs to the optimal solution. Considered values include $k = 2$, $k = 3$ and $k = 4$ (forming non-deceptive, quasi-deceptive and deceptive problems respectively). Studying the scalability for different settings of k can offer a better understanding of the model since not only the scalability is analysed but also how the scalability changes when the problem difficulty increases.

There are two distinct regions in the search space of trap functions, one leading to a global optimum and the other leading to the local optimum (see Figure 1.1). In general, a trap function is defined by the following equation:

$$trap(u(\overrightarrow{x})) = \begin{cases} \frac{a}{z}(z - u(\overrightarrow{x})), & \text{if} \quad u(\overrightarrow{x}) \leq z \\ \frac{b}{l-z}(u(\overrightarrow{x}) - z), & \text{otherwise} \end{cases} \tag{1.1}$$

where $u(\overrightarrow{x})$ is the unitation function, a is the local optimum, b is the global optimum, l is the problem size and z is a slope-change location separating the attraction basin of the two optima.

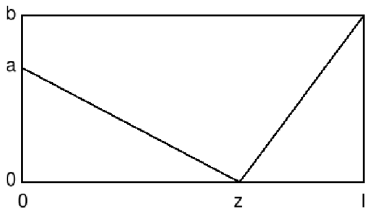

Figura 1.1: Generalised *l-trap* function.

For the following experiments, 2-trap, 3-trap and 4-trap functions were designed with the following parameter values: $a = l - 1$, $b = l$, and $z = l - 1$. With these settings[1], 2-trap is not deceptive, 4-trap is deceptive and 3-trap lies in the region between deception and non-deception. Under these conditions, it is possible not only to examine the scalability on trap functions, but also to investigate how the scalability varies when changing from non-deceptive to deceptive search landscapes. Scalability tests were performed by juxtaposing m trap functions in binary strings of length L and summing the fitness of each sub-function to obtain the total fitness.

1.3.2. A method for estimating the population size

Kumara Sastry proposes in [Sas01] a method based on bisection to estimate the optimal population size N to solve a problem instance, that is, the lowest N for which 98 % of the runs find the problem optimum. To this end, a selectorecombinative GA is used to search the minimum population size such that using random initialisation it is able to converge to the optimum without any other mechanism than recombination and selection.

Algorithm 1 depicts the method based on bisection. The method begins with a small population size which is doubled until the algorithm ensures a reliable convergence. We define the reliability criterion as the convergence of the algorithm to the optimum 49 out of 50 times. After that, the interval (min, max) is halved several times and the population size adjusted within

[1]Originally, Ackley's trap functions use $z = \frac{3l}{4}$, however, [DG91] demonstrates that trap functions are fully easy under such settings.

Algorithm 1 Population tuning algorithm based on bisection

N = Initial Population Size (20)
while GA reliability (N) $< 98\,\%$ **do**
  $min = N; max, N = 2N$
end while
while $\frac{max-min}{min} > \frac{1}{16}$ **do**
  $N = \frac{max+min}{2}$
  if GA reliability(N) $< 98\,\%$ **then**
    $min = N$
  else
    $max = N$
  end if
end while

such a range until $\frac{max-min}{min} > threshold$, where min and max stand respectively for the minimum and maximum population size estimated and *threshold* for the accuracy of the adjustment within such a range. This parameter has been set to $\frac{1}{16}$ in order to obtain a good adjustment of the population size.

1.3.3. Metrics

The following metrics will be used for assessing the performance of the model in the experimental analysis. To allow the comparison of results with those in the literature such metrics were choosed to be standard in EAs. Eiben and Smith make an extensive revision of them in [ES03] or, in addition, Tomassini in [Tom05].

- The success rate (SR) measures the algorithm quality as the proportion in which the algorithm is able to find the problem optimum out of all the runs.
- The average number of evaluations to solution (AES) stands for the number of evaluations that the algorithm spends in those runs that yield success. Since a preliminary analysis on the normality of results shows that they do not follow a normal distribution, we have chosen the number of evaluations to solution in the third quartile ($AESQ_3$) as value of reference, meaning that a $75\,\%$ of the runs will stay below such a value.
- The Mean Best Fitness (MBF) is used to depict the algorithm convergence as the averaged values of the best fitness.

- The genotypic entropy (GE) is a measure of the population diversity defined on the genotypic distances ($H_g(P)$).

$$H_g(P) = -\sum_{j=1}^{N} g_j log(g_j) \tag{1.2}$$

where g_j is the fraction $\frac{n_j}{N}$ of individuals in P having a Hamming distance j to the optimal genotype, and N is the number of different distances.

1.3.4. Summary

This section presents the experimental methodology that will be followed for analysing the viability of the EvAg approach. To that aim, we will have to prove that the algorithm is *scalable* and *fault-tolerant* in a simulated P2P environment.

In order to tackle such goals, we will perform an experimental analysis of the three test-cases summarised in Table 1.1. In every case, the analysis will follow the same methodology consisting in the study of the algorithmic scalability, the analysis of the algorithm convergence and population diversity at run-time and a comparison of the results to show whether they present statistical differences or not.

Test-Case	Methodology
1 Scalability in a failure-free environment vs. sequential GAs	1.1 Scalability selectorecombinative GA 1.2 Analysis of the convergence of the larger problem instance 1.3 Statistical analysis
2 Influence of the population structure on the algorithm performance	2.1 Scalability selectorecombinative GA 2.2 Analysis of the convergence of the larger problem instance 2.3 Statistical analysis
3 Fault tolerance of the model	3.1 Scalability selectorecombinative GA 3.2 Analysis of the convergence under different churn scenarios 3.3 Statistical analysis

Cuadro 1.1: Summary of the experimental methodology.

1.4. Test-Case 1: Scalability of the model in failure-free environments

In order to investigate the scalability of the EvAg model, experiments were conducted on different trap functions and compared against two canonical GAs, a steady-state GA (SSGA) and a generational GA (GGA). Whereas the population structure of the EvAg model is defined by the newscast protocol, the canonical GAs are panmictic. Following the experimental methodology of Section 1.3, two series of experiments were conducted:

In the first series, experiments use selectorecombinative versions of the GAs to estimate optimal population sizes for the different problem instances. The reason for using selectorecombinative GAs is that there are well defined models to establish the population size and the number of evaluations required to solve a given trap function instance [HCPGM99], meanwhile, to the best of our knowledge there are no such models when using mutation. To this end, in Section 1.3.2 we have described a method for estimating the population size. The underlying idea is that without mutation, the population size scales with respect to the problem instance since it becomes the only source of diversity. In this context, Thierens demonstrates in [Thi99] the necessity of larger population sizes when tackling larger problem instances.

In the second series, the analysis focuses on the performance of the different approaches when they are equally parameterized. Specifically, large instances of 2-Trap, 3-Trap and 4-Trap are considered to analyse the convergence of the fitness and the evolution of genotypic diversity. Finally, such results are statistically analysed in order to determine whether the difference in performances are significant or not.

1.4.1. Scalability analysis

In this first series of experiments, different approaches have no mutation in order to meet the selectorecombinative criterion of the bisection method. All settings are summarised in Table 1.2 taking into account decisions in Section 1.2 about choosing operators that do not take advantage of the problem structure or setting the cache size. Since the methodology imposes a SR of 0.98 in the results, the $AESQ_3$ has been used as an appropriate metric to measure the computational effort to reach the success criterion. A more efficient algorithm will need a smaller number of evaluations.

Figure 1.2 depicts the scalability of the population size (N) and the required number of evaluations to reach the problem optimum ($AESQ_3$) for the three approaches under study on 2, 3, and 4-trap functions. All the graphics

Trap instances	
BB size	$2, 3, 4$
Individual Length (L)	$12, 24, 36, 48, 60$
GA settings	
GA	selectorecombinative SSGA
	selectorecombinative GGA
	selectorecombinative EvAg
Population size	Tuning algorithm
Selection of Parents	Binary Tournament
Recombination	Uniform crossover, $p_c = 1{,}0$
Newscast settings	
Cache size	20

Cuadro 1.2: Test-case 1: Parameters of the experiments in the analysis of scalability.

show that either the population size or the computational effort fit with a potential order of scalability with base the length of the chromosome (L) and different exponents depending on the problem difficulty and the approach itself.

The first conclusion that can be easily drawn from results is a better scalability of the EvAg approach with respect to the population size, specially when the problem difficulty increases from 2 to 4-trap. That is, increasing the problem difficulty makes that the GGA and SSGA face extreme difficulties to track the problem optimum, thus requiring a higher population size N to prevent that the algorithm gets stuck in local optima. From a computational perspective, this fact can be translated into a more efficient use of the running platform since the EvAg approach will require a smaller amount of computational resources. Additionally, results on $AESQ_3$ are clearly correlated to the population size, SSGA scales better than GGA and it is roughly similar to EvAg in 2 and 3-trap. Nevertheless, as the problem difficulty increases to 4-trap, EvAg scales clearly better.

In order to gain some insight on the influence of mutation in the scalability order, we consider a more realistic GA setup and switch mutation on. This implies that we have to specify values for the mutation rate parameter p_m. Strictly speaking, we should also recalibrate population sizes, since the bisection method only gives good estimates for selectorecombinative GAs. However, an extensive parameter sweep goes far beyond the scope of this thesis and therefore we will use the common "$\frac{1}{L}$ heuristic" for setting the mutation rates and keep the population sizes that were used in the first series of experiments. Obviously, in this case we only need to look at the $AESQ_3$ results. The outcomes of these experiments are shown in Figure 1.3 in which curves appear shifted with respect to the selectorecombinative version in

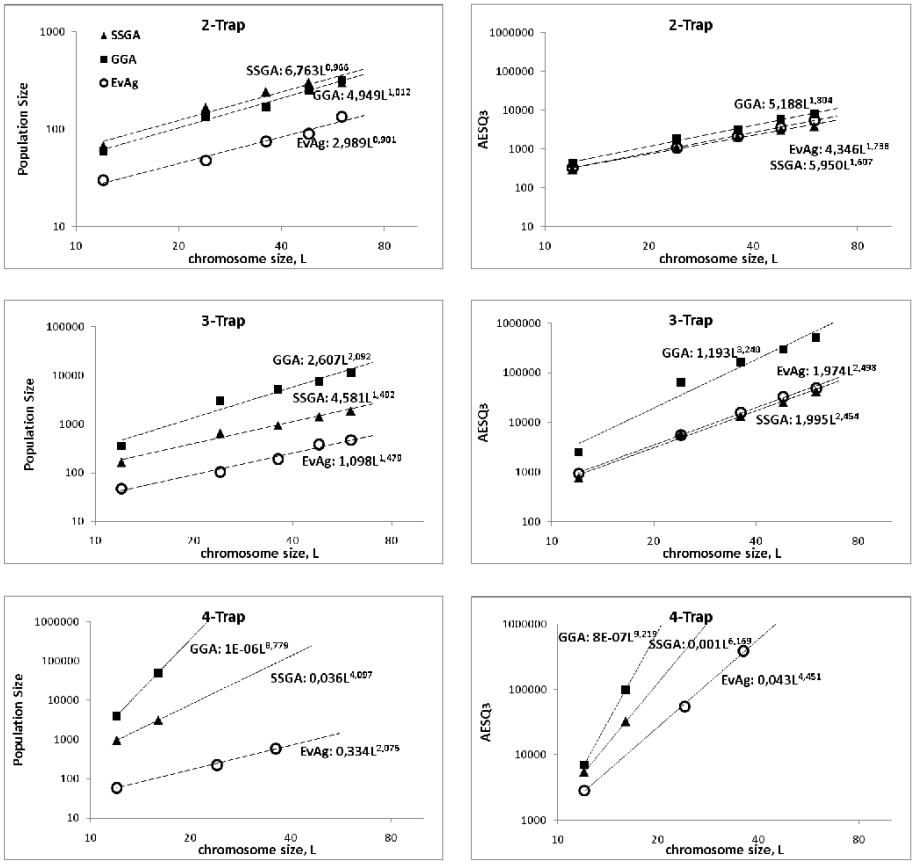

Figura 1.2: Scalability in trap functions based on the population tuning algorithm and the selectorecombinative versions of the generational GA (GGA), steady-state GA (SSGA) and the Evolvable Agent (EvAg). On the left the estimated population sizes N and the evaluations to solution in third quartile $AESQ_3$ on the right. Results are obtained by bisection and depicted in a *log-log* scale as a function of the length of the chromosome, L.

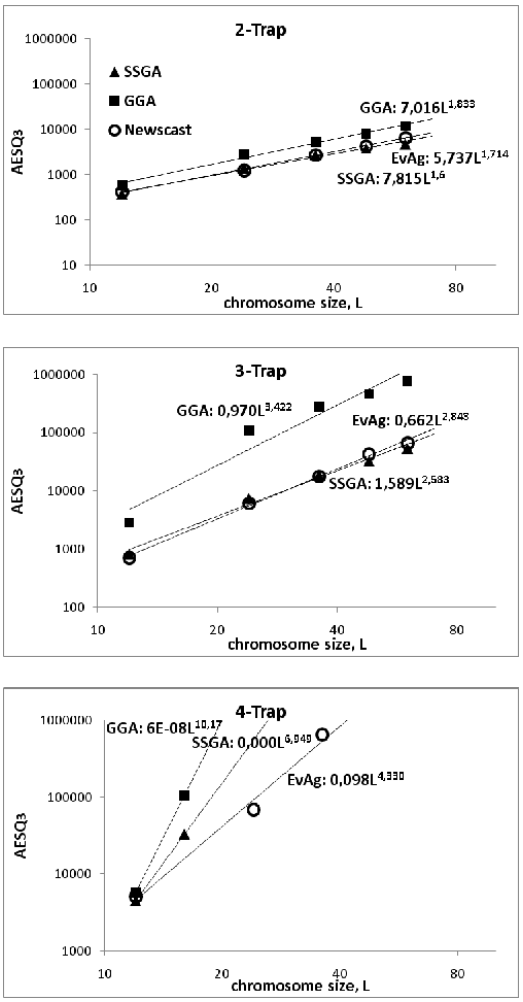

Figura 1.3: Reproduction of the results in Figure 1.2 with mutation switched on.

Figure 1.2 but approximately keeping the same scalability order. Table 1.3 compares such estimated complexity orders with mutation switched off and on, showing that mutation does not alter the order in algorithm performance, and exponents are roughly the same, with only the constant in the power law changing.

	GGA Mutation		SSGA Mutation		EvAg Mutation	
	off	on	off	on	off	on
2-Trap	$O(L^{1,804}))$	$O(L^{1,833})$	$O(L^{1,607})$	$O(L^{1,6})$	$O(L^{1,738})$	$O(L^{1,714})$
3-Trap	$O(L^{3,24})$	$O(L^{3,422})$	$O(L^{2,454})$	$O(L^{2,583})$	$O(L^{2,498})$	$O(L^{2,843})$
4-Trap	$O(L^{9,219})$	$O(L^{10,17})$	$O(L^{6,169})$	$O(L^{6,949})$	$O(L^{4,451})$	$O(L^{4,33})$

Cuadro 1.3: Complexity orders of the $AESQ_3$ scalability in O notation.

1.4.2. Analysis on the algorithm convergence

In the second series of experiments we try to gain more detailed insights in the differences between the three approaches to population management using mutation. To this end, we run the GGA, SSGA, and EvAg models using the same settings on a problem instance as summarised in Table 1.4 (recall, that in the previous experiments, GGA, SSGA, and EvAg used different population sizes on any given problem instance). We choose large instances of each problem under study for this purpose (i.e. $L = 36$ in 4-trap, $L = 99$ in 3-trap and $L = 400$ in 2-trap), where the differences in scalability between the approaches are most visible.

To estimate the population sizes and the maximum number of evaluations in each case, we have used the scalability orders for the selectorecombinative EvAg on the previous series, e.g. $L = 99$ in 3-trap will require a population size of $1{,}098L^{1,479} = 942$ and a maximum number of evaluations of $1{,}974L^{2,498} = 190740$. With these settings, switching mutation on implies that such values are oversized for the EvAg approach since mutation represents a new source of diversity, however, the highest scalability orders of SSGA and GGA indicate that such population sizes will remain undersized for both approaches.

The results in Figure 1.4 show that the convergence of the SSGA and GGA approaches stagnates at early stages of the search in the different instances. Despite the SSGA performing a more exploitative search than the GGA, as shown by the genotypic entropy converging to zero, both cases lost track of the optimum, a fact that might be explained by an undersized population size. On the other hand, the evolution of diversity of the EvAg approach

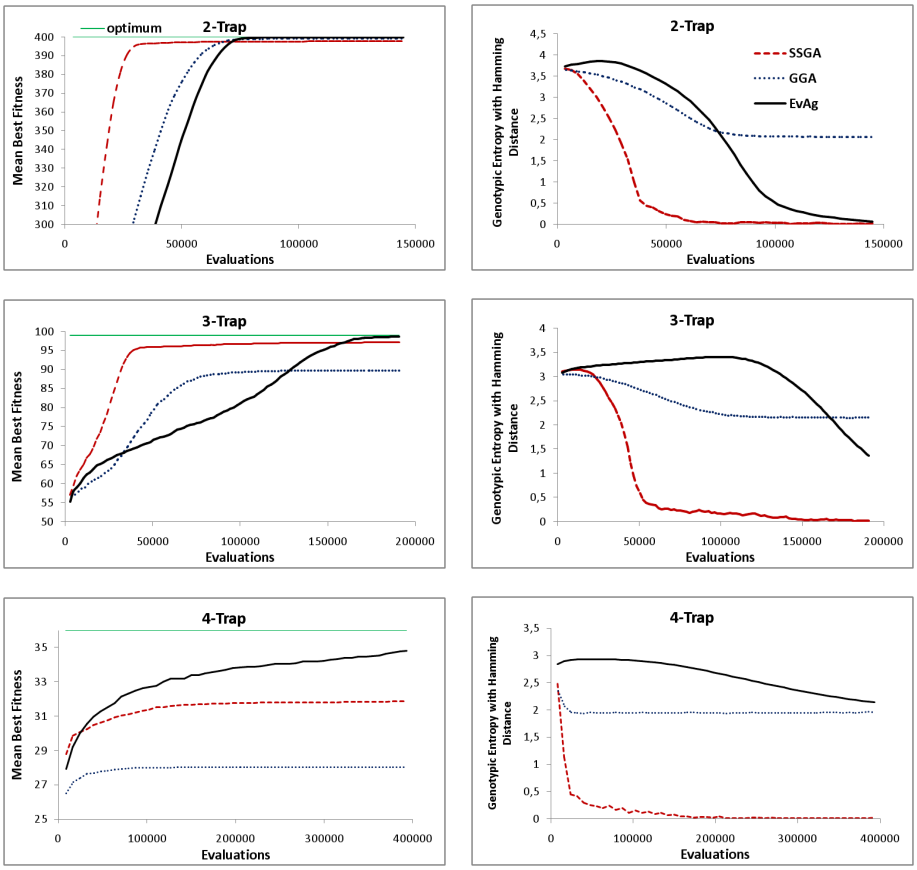

Figura 1.4: Test-case 1: Best fitness convergence(*left*) and evolution of the diversity expressed as the entropy based on the hamming distances between the genotypes(*right*). Graphs plotted represent the average of 50 independent runs.

Trap instances	
2-Trap	
Individual Length (L)	400
Population size	660
Termination Condition	Max. Eval. =144700
3-Trap	
Individual Length (L)	99
Population size	942
Termination Condition	Max. Eval. =190740
4-Trap	
Individual Length (L)	36
Population size	600
Termination Condition	Max. Eval. =393000
GA settings	
GA	SSGA
	GGA
	EvAg
Selection of Parents	Binary Tournament
Recombination	Uniform crossover, $p_c = 1{,}0$
Mutation	Bit-flip mutation, $p_m = \frac{1}{L}$
Newscast settings	
Cache size	20

Cuadro 1.4: Test-case 1: Parameters of the experiments for the analysis of convergence.

indicates that the algorithm converges to the optimum in 2 and 3-trap and is still converging in 4-trap when the termination criterion is met. In this last case, it is straightforward to see that the population size is oversized since a selectorecombinative EvAg is able to find the optimum in such number of evaluations. Given that the three approaches are equally parameterized, it is within the spatially structured scheme of reproduction of the EvAg model where the genetic diversity is preserved at a higher level and consequently the population size N can be reduced. Hence, the EvAg approach scales better on difficult problems. With a lower optimal N, EvAg needs fewer evaluations to reach the optimum when compared to panmictic GAs.

1.4.3. Statistical analysis

Previous results are pre-analysed in Table 1.5 using the Anderson-Darling test to refutate the null hypothesis on the normality of the data. The small p-values at the best fitness distributions show that results do not follow a normal distribution. This way, a non-parametric Wilcoxon test is used to compare the quality of fitness between the EvAg and the SSGA and GGA approaches.

Table 1.6 presents the Wilcoxon analysis of the data showing significant differences between the EvAg and canonical approaches in all the problem

Problem Instance	*Algorithm*	*Anderson-Darling*	*Normal distribution?*
Trap2	GGA	A=3.5 p-value=6.1e-09	no
	SSGA	A=1.8 p-value=6.6e-05	no
	EvAg	A=17 p-value<2.2e-16	no
Trap3	GGA	A=0.9 p-value=0.01	no
	SSGA	A=1.8 p-value=9.9e-05	no
	EvAg	A=11.7 p-value<2.2e-16	no
Trap4	GGA	A=2.4 p-value=2.035e-06	no
	SSGA	A=1.2 p-value=0.002	no
	EvAg	A=12 p-value<2.2e-16	no

Cuadro 1.5: Anderson-Darling test on the normality of the best fitness distributions. Results are obtained over 50 independent runs.

instances. Therefore, it can be concluded that EvAg outperforms SSGA and GGA when tackling large instances of trap functions.

Problem Instance	*Algorithm*	*Avg. Fitness* $\pm\sigma$	*Wilcoxon Test*	*Significantly different?*
Trap2	GGA	399.08±1.01	W=624 p-value=1.637e-07	yes
L=400	SSGA	397.8±1.56	W=142 p-value<2.2e-16	yes
N=660				
M. Eval.= 144700	EvAg	**399.92±0.27**	-	-
Trap3	GGA	89.78±2.8	W=0 p-value<2.2e-16	yes
L=99	SSGA	97.16±1.46	W=402 p-value=3.623e-10	yes
N=942				
M. Eval.= 190740	EvAg	**98.72±0.60**	-	-
Trap4	GGA	28.02±0.14	W=2500 p-value <2.22e-16	yes
L=36	SSGA	31.88±1.15	W=2476 p-value<2.22e-16	yes
N=600				
M. Eval.= 393000	EvAg	**34.82±0.66**	-	-

Cuadro 1.6: Wilcoxon test comparing the best fitness distributions of equally parameterized SSGA, GGA and EvAg in 2,3 and 4-trap. Results are obtained over 50 independent runs.

1.4.4. Summary

The EvAg model has demonstrated good scalability on trap functions with search landscapes of different difficulties. We found that EvAg scales better than a GGA and an SSGA that only differ from it in the population structure. Based on various scenarios with and without mutation we can conclude that EvAg needs fewer evaluations to reach a solution in addition to requiring smaller populations. The improvement is much more noticeable as the problem difficulty increases showing thereby the adequacy of the P2P approach for tackling large instances of difficult problems. These results are specially remarkable in the deceptive case for a BB size of 4. It illustrates the good scalability of the parallel approach when tackling difficult problems.

To gain more detailed insights, the runtime behavior of equally parameterized EvAg , SSGA, and GGA approaches has been analysed on large problem instances where the differences of scalability are more outstanding. The results show that EvAg can maintain higher population diversity and better progress in fitness. As a consequence, an oversized population for the EvAg

model still remains undersized for the canonical approaches that get lost in local optima.

1.5. Test-Case 2: Influence of the population structure on the algorithm performance

As described in Section **??**, the EvAg model impose no restrictions on the choice of a population structure. Within that context, the newscast protocol has been considered throughout this thesis so that it allows a decentralised execution of the approach in a P2P system. Nevertheless, different structures will influence the dynamics of the algorithm in a different way and therefore, its algorithmic performance. Hence, the aim of this test-case is the comparison of performances of the approach using different population structures as newscast, a ring and a Watts-Strogatz (depicted in Figure 1.5).

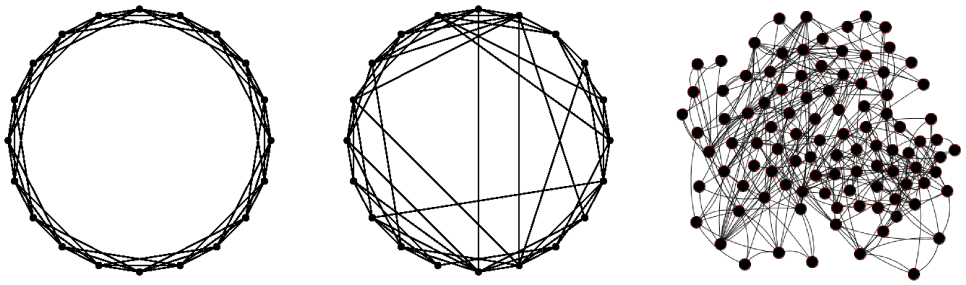

Figura 1.5: From left to right: ring, Watts-Strogatz and newscast population structures.

We have chosen a ring population structure as an instance to compare regular lattices performances against newscast since most of the works in the literature considering fine-grained approaches use to focus in regular lattices (some examples have been already mentioned in this thesis including [GS89, AD08, Tom05, DAGT04a, GATT04, GTTA05]). The main reason for such a choice is that the bisection method is not applicable for some other regular lattices such as a toroid or a grid in which the square dimensions of the topology do not allow doubling the size of the population in the process of estimating the correct size.

In addition, the Watts-Strogatz method (described in Section **??**) represents an easy and understandable model for creating a small-world population structure. This way, it will be possible to compare two different methods (i.e. Watts-Strogatz and newscast) for generating the same sub-type of complex network. The interest here goes a step further than in the case of the ring since there are many P2P protocols designed to work as small-world networks. Therefore, we aim to establish whether the properties on scalability of the newscast population structure lie in its small-world structure so that

may extend to other protocols implementing the same kind of topologies (e.g. Gnutella 0.4 [GDF01] or any DHT [SMK+01, RD01, ZKJ01, RFH+01]).

1.5.1. Scalability analysis

The folowing experiment aims to compare the influence of different kind of decentralised population structures on the scalability of the EvAg model when tackling trap functions. Table 1.7 summarises the settings in these series including the ring, Watts-Strogatz and newscast as population structures.

Trap instances	
BB size	$2, 3, 4$
Individual Length (L)	$12, 24, 36, 48, 60$
GA settings	
GA	selectorecombinative EvAg
Population size	Tuning algorithm
Selection of Parents	Binary Tournament
Recombination	Uniform crossover, $p_c = 1{,}0$
Population settings	
Population structure	Ring
	Watts-Strogatz
	Newscast
Node degree	20

Cuadro 1.7: Test-case 2: parameters of the experiments in the analysis of scalability.

Figure 1.6 depicts the scalability of the population size (N) and the computational effort ($AESQ_3$) for the two approaches under study in 2, 3 and 4-trap functions...

Results show that...

1.5.2. Analysis on the algorithm convergence

This section analyses the convergence of the EvAg approach in 2, 3 and 4-trap instances for different degradation rates $\lambda = 400, 2500$ with respect to a failure-free execution. All settings are summarised in Table 1.8.

1.5.3. Statistical analysis

In order to provide previous results with an adequate statistical processing, Table 1.12 presents the Wilcoxon analysis of the data showing significant differences between the fitness distributions for the EvAg model with and without failures in the system.

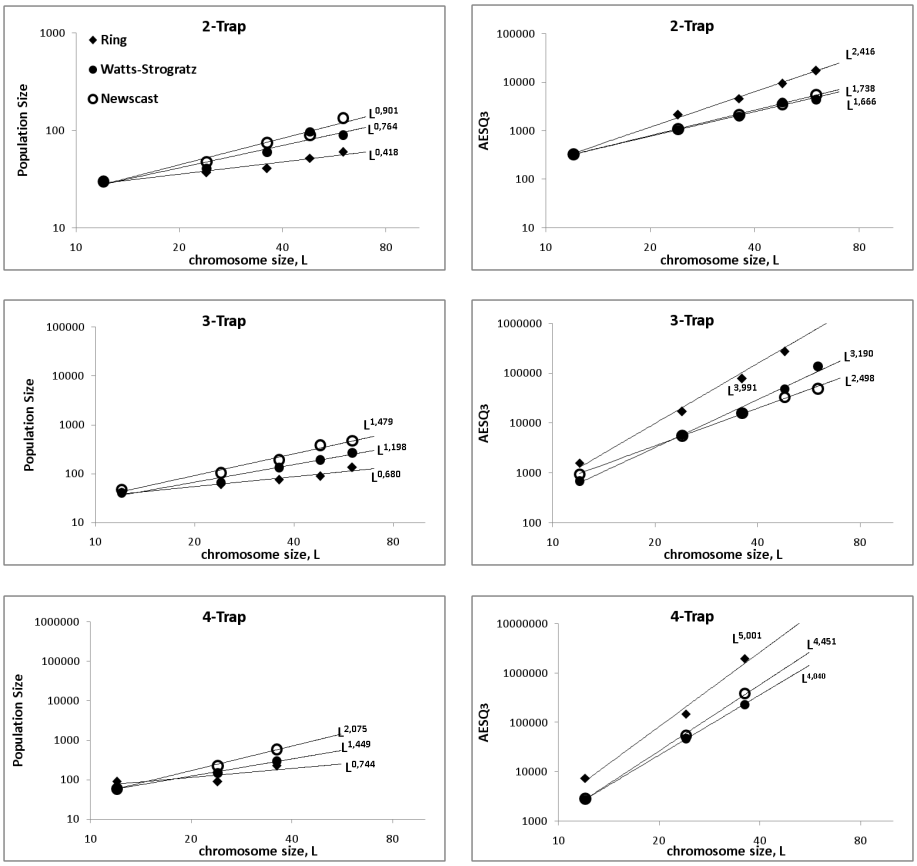

Figura 1.6: Scalability of the EvAg model using a Ring, Watts-Strogatz and Newscast population structures in trap functions for the estimated population sizes N *(left)* and the evaluations to solution in third quartile *(right)*. Results are obtained by bisection and depicted in a *log-log* scale as a function of the length of the chromosome, L.

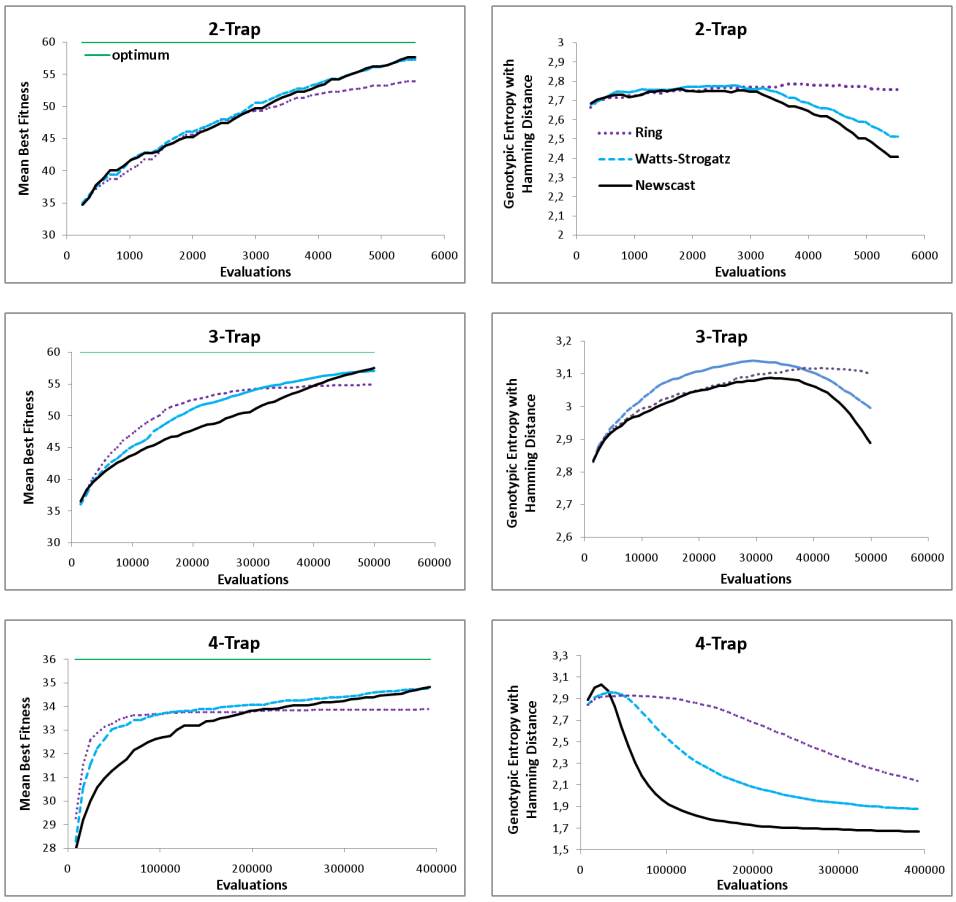

Figura 1.7: Test-Case 2: Best fitness convergence(*left*) and evolution of the diversity expressed as the entropy based on the hamming distances between the genotypes(*right*). Graphs plotted represent the average of 50 independent runs.

Trap instances	
2-Trap	
Individual Length (L)	60
Population size	135
Termination Condition	Max. Eval. = 5535
3-Trap	
Individual Length (L)	60
Population size	480
Termination Condition	Max. Eval. =49920
4-Trap	
Individual Length (L)	36
Population size	600
Termination Condition	Max. Eval. =393000
GA settings	
GA	EvAg
Selection of Parents	Binary Tournament
Recombination	Uniform crossover, $p_c = 1{,}0$
Mutation	Bit-flip mutation, $p_m = \frac{1}{L}$
Population settings	
Population structure	Ring
	Watts-Strogatz
	Newscast
Node degree	20

Cuadro 1.8: Test-Case 2: Parameters of the experiments for the analysis of convergence.

The statiscal analysis confirms the conclusion on the system degradation outperforming the failure-free run. In other words, the EvAg model is inherently fault-tolerant and degrades gracefully.

Problem Instance	*Algorithm*	*Avg. Fitness* $\pm\sigma$	*Wilcoxon Test*	*Significantly different?*
Trap2				
L=60	Ring	53.88±1.21	W=2429 p-value=2.22e-16	yes
N=135	**Watts-Strogatz**	**57.26±1.52**	**W=1386 p-value=0.335**	**no**
M. Eval.= 5535	**Newscast**	**57.6±1.38**	-	-
Trap3				
L=60	Ring	54.9±0.97	W=2135 p-value=6.46e-10	yes
N=480	**Watts-Strogatz**	**57.04±1.07**	**W=1536 p-value=0.045**	**yes**
M. Eval.= 49920	*Newscast*	*57.5±2.04*	-	-
Trap4				
L=36	Ring	33.88±0.69	W=2044 p-value=4.38e-09	yes
N=600	**Watts-Strogatz**	**34.78±0.65**	**W=1288 p-value=0.77**	**no**
M. Eval.= 393000	**Newscast**	**34.82±0.66**	-	-

Cuadro 1.9: Wilcoxon test comparing the best fitness distribution of the EvAg model using a Ring, Watts-Strogatz and Newscast population structures. Results are obtained over 50 independent runs.

1.5.4. Summary

1.6. Test-Case 3: Fault tolerance of the model

As explained in chapter **??**, P2P systems are large networks of volatile resources in which the collective dynamics of peers joining and departing from the system are known as *churn*. This way, addressing *churn* in a P2P EA turns into a requirement of design since failures in peers are inherent to the system.

Following the work by Stutzbach and Rejaie in [SR06], there are two main group-level properties of *churn* characterising the behaviour of every participating peer: The *inter-arrival time* and the *session length*, respectively, the time between two sessions and the time from the beginning to the end of a session. In this test-case, we have assumed that all peers start at the same time with a certain *session length* and avoiding *inter-arrivals*. Therefore, once that a peer leaves the system, it does not re-join again so that the system *degrades* from the initial configuration.

The *session length* can be modeled randomly from a Weibull distribution using the following formula:

$$X = \lambda(-ln(U))^{\frac{1}{k}} \tag{1.3}$$

where U is drawn from the uniform distribution, k stands for the shape of the *degradation* and λ for the time scale.

In this context, Stutzbach and Rejaie analyse the runtime dynamics of three real P2P systems and conclude that all *session lengths* fit with a Weibull distribution of shape $k \approx 0{,}40$ but differing on the time dimension λ. Given that experiments are conducted in a simulator and a simulator cycle can apply for different time units in real time, λ parameter was pre-adjusted to simulate different failure rates in such a way that the system degrades up to 90 % in the worst case. In concrete, we use the following values for $\lambda = 400, 2500$ depicted in Figure 1.8. It shows the Weibull cumulative distribution functions for such values, representing the percentage of remaining nodes at each moment of a experiment (e.g. in the cycle 2000, ∼15 % of the peers remain for $\lambda = 400$ and ∼75 % for $\lambda = 2500$).

1.6.1. Scalability analysis

Using previous *degradation* rates, several instances of 2, 3 and 4-trap have been analysed in order to assess the impact of *churn* in the EvAg model. All settings for the experiments are summarised in Table 1.10.

This way, there will be three variables affecting the performance of the algorithm: The size of the problem (L), which will conduct to a scalability

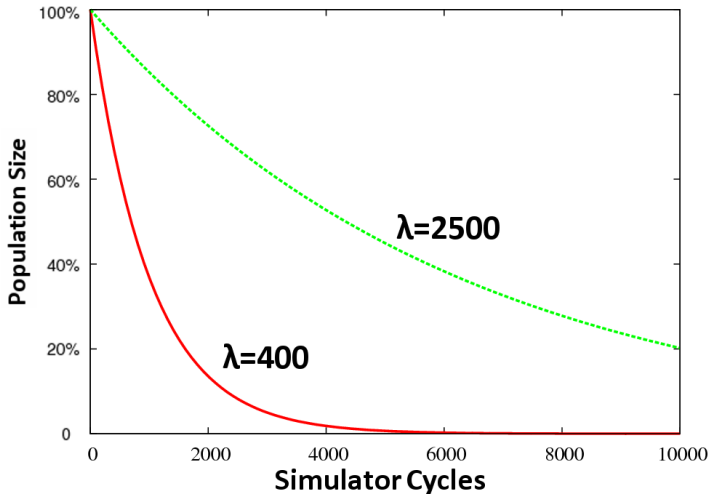

Figura 1.8: Complementary cumulative distribution functions for a Weibull distribution in function of the simulator cycles. Percentages are obtained as the ratio between the total number of failures and the initial population size.

analysis, the intensity of *churn* (λ) and the initial[2] population size (N). Being λ and L two independent variables under the condition of obtaining a SR of 0.98, the initial population size can be expressed as a function $f(\lambda, L) = N$ and empirically estimated using the bisection method.

Trap instances	
BB size	$2, 3, 4$
Individual Length (L)	$12, 24, 36, 48, 60$
GA settings	
GA	selectorecombinative EvAg
Population size	Tuning algorithm
Selection of Parents	Binary Tournament
Recombination	Uniform crossover, $p_c = 1{,}0$
Newscast settings	
Cache Size	20
Scenarios of churn	
λ	400,2500
k	0.4

Cuadro 1.10: Test-case 3: parameters of the experiments in the analysis of scalability.

Figure 1.9 shows the scalability of the population size (N) as a function of

[2]Given that the system degrades, the initial population size will not correspond to the final one.

L, that is, L scales and λ remains fixed in $f(\lambda, L) = N$. *Churn* does not seem to damage the scalability of the approach since estimated curves $f(\lambda, L)$ roughly follow the same orders of scalability and just appear shifted by a constant which is *churn* dependent. The more intensive the churn, the bigger the constant. In this sense, a small increase on the initial population size is enough to provide resilience to system failures since orders of scalability go from $O(L^{0,901})$ to $O(L^{0,928})$ in 2-trap, $O(L^{1,479})$ to $O(L^{1,799]})$ in 3-trap and are estimated to $O(L^{2,075})$ for any scenario in 4-trap.

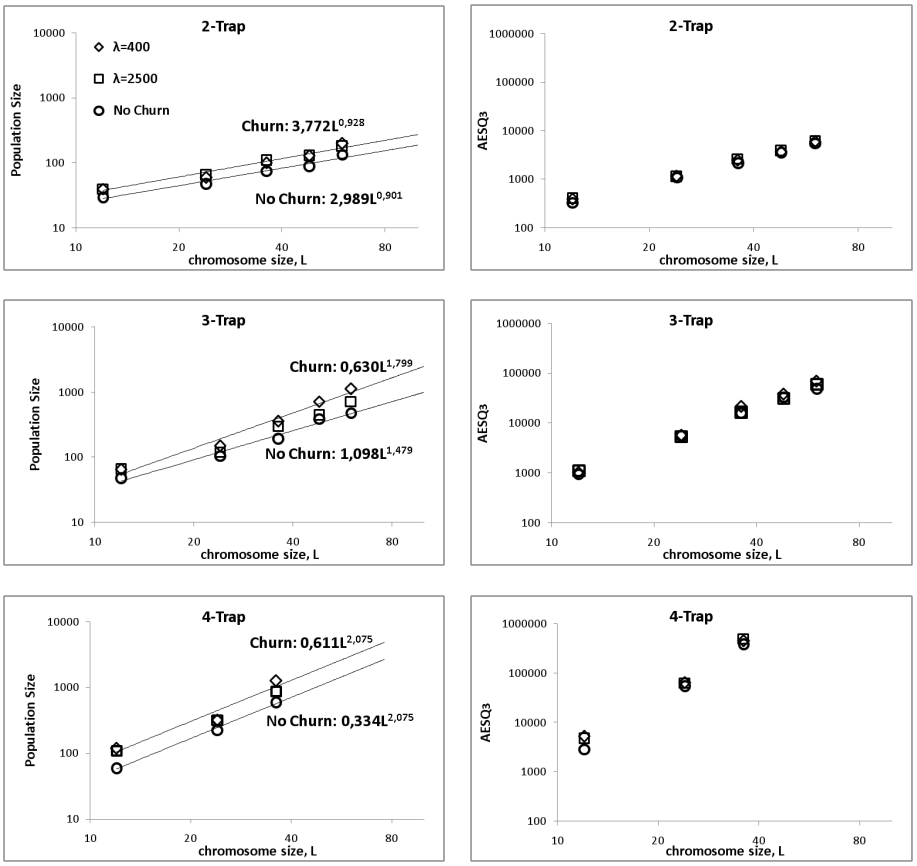

Figura 1.9: Scalability of the EvAg model for two different scenarios of churn (λ) and a failure-free environment in trap functions for the estimated population sizes N *(left)* and the evaluations to solution in third quartile *(right)*. Results are obtained by bisection and depicted in a *log-log* scale as a function of the length of the chromosome, L.

In addition, results on the $AESQ_3$ show that the computational efforts required for tackling any given instance are independent from the *churn* scenario. Given that *churn* does not affect the scalability of the computational

efforts, they exclusively depend on the problem instance size L and, therefore, EvAg degrades gracefully. We will require the same computational efforts under any *churn* scenario if we ensure enough resources to satisfy the condition of a SR of 0.98.

Figure 1.10 provides a better idea to the extent of these results. It represents the percentage of individuals of N for which each experiment is expected to end. The effects of *churn* are more pernicious as the instances scale. In the worst case (i.e. $L = 36$ in 4-trap and $\lambda = 400$), the initial population ends with a $\sim 10\,\%$ of the individuals, still guaranteeing a reliable convergence.

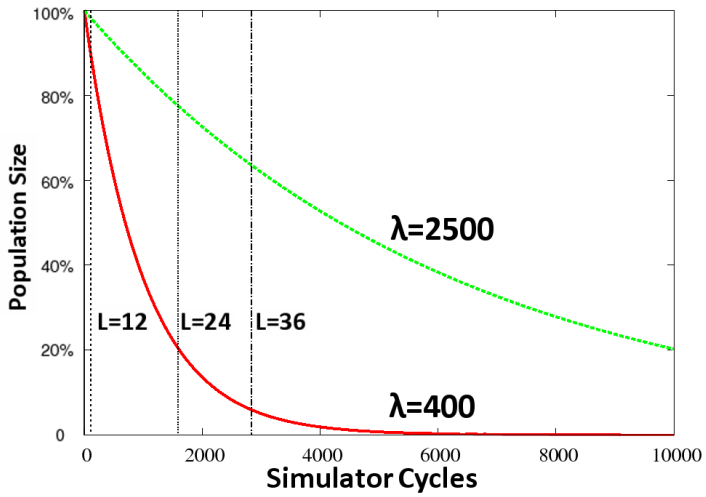

Figura 1.10: Degradation of the system for the different failure rates and execution times in three instances of the 4-trap function.

1.6.2. Analysis on the algorithm convergence

This section analyses the convergence of the EvAg approach in 2, 3 and 4-trap instances for different degradation rates $\lambda = 400, 2500$ with respect to a failure-free execution. All settings are summarised in Table 1.11.

Initial population sizes have been set to those values obtained in the previous series for the failure-free run of the selectorecombinative EvAg . Using such values means that populations will be oversized for the failure-free counterpart in these series that use mutation. Nevertheless, that might not be the case when the system degrades. As previously seen, a small increase on the initial population size is sufficient condition for tolerating faults. Given

that populations are oversized for a failure-free run using mutation, values will approximate an optimal size whenever the system degrades.

Trap instances	
2-Trap	
Individual Length (L)	60
Population size	135
Termination Condition	Max. Eval. = 5535
3-Trap	
Individual Length (L)	60
Population size	480
Termination Condition	Max. Eval. =49920
4-Trap	
Individual Length (L)	36
Population size	600
Termination Condition	Max. Eval. =393000
GA settings	
GA	EvAg
Selection of Parents	Binary Tournament
Recombination	Uniform crossover, $p_c = 1{,}0$
Mutation	Bit-flip mutation, $p_m = \frac{1}{L}$
Newscast settings	
Cache size	20
Scenarios of churn	
λ	400,2500
k	0.4

Cuadro 1.11: Test-Case 3: Parameters of the experiments for the analysis of convergence.

Figure 1.11 shows indeed that the EvAg model converges better when the system degrades, specially in the worst case for $\lambda = 400$. This fact is remarkable since the degradation of the system do not inflict a penalisation in the quality of solutions as could be expected, but, it is able to outperform the failure-free run despite EvAgs departing possibly contain valid solutions.

Besides, genetic diversity decreases faster as the system degradation turns more intensive which can be translated into a more exploitative behavior of the algorithm. Such conclusion points out a possible explanation for the fault-tolerance of the model; as it has already been shown in Sections 1.4 and 1.5, EvAg is good at the preservation of genetic diversity and this way, it is able to balance the effect of an increasing explotation component when the system degrades.

Finally and providing a cuantitative view on the run-time dynamics of *churn*, Figure 1.12 depicts the degradation of the population for the different runs in the 4-trap instance. It shows how the system degrades up to $\sim 90\,\%$ in the worst case for $\lambda = 400$.

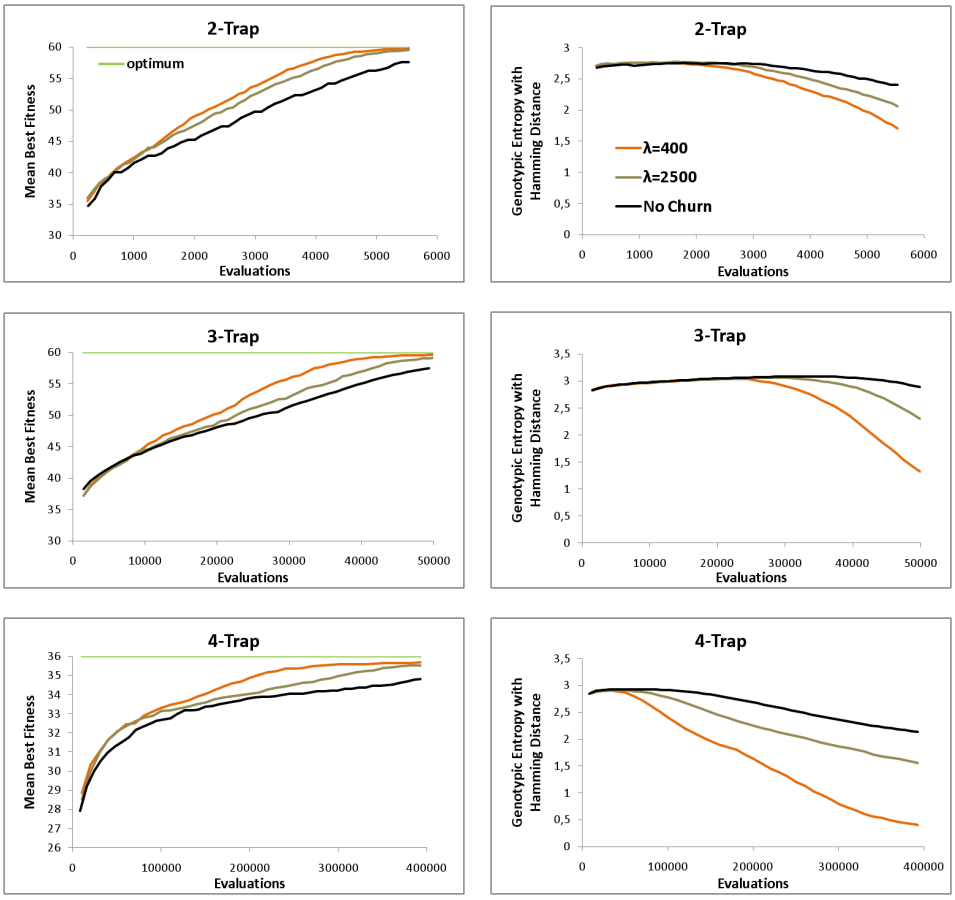

Figura 1.11: Test-case 3: Best fitness convergence(*left*) and evolution of the diversity expressed as the entropy based on the hamming distances between the genotypes(*right*). Graphs plotted represent the average of 50 independent runs.

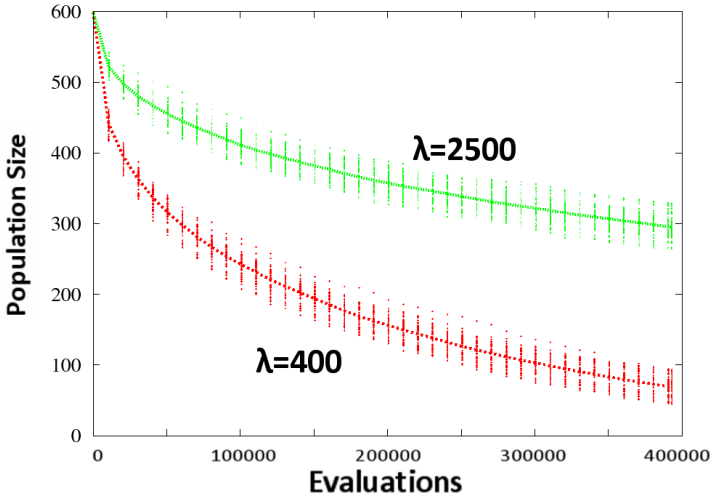

Figura 1.12: Population dynamics in the 4-trap instance for $L = 36$. Population sizes of the different runs are represented as points and the respective promediated values as lines for $\lambda = 400$ and $\lambda = 2500$.

1.6.3. Statistical analysis

In order to provide previous results with an adequate statistical processing, Table 1.12 presents the Wilcoxon analysis of the data showing significant differences between the fitness distributions for the EvAg model with and without failures in the system.

The statiscal analysis confirms the conclusion on the system degradation outperforming the failure-free run. In other words, the EvAg model is inherently fault-tolerant and degrades gracefully.

Problem Instance	*Churn*	*Avg. Fitness* $\pm\sigma$	*Wilcoxon Test*	*Significantly different?*
Trap2				
L=60	$\lambda = 400$	**59.82±0.43**	**W=184 p-value=6.32e-15**	**yes**
N=135	$\lambda = 2500$	59.5±0.78	W=320 p-value=2.9e-11	yes
M. Eval.= 5535	*No Churn*	*57.6±1.38*	-	-
Trap3				
L=60	$\lambda = 400$	**59.62±0.62**	**W=414 p-value=1.37e-9**	**yes**
N=480	$\lambda = 2500$	59.14±1.2	W=614 p-value=6.2e-6	yes
M. Eval.= 49920	*No Churn*	*57.5±2.04*	-	-
Trap4				
L=36	$\lambda = 400$	**35.68±0.46**	**W=447 p-value=1.89e-9**	**yes**
N=600	$\lambda = 2500$	35.54±0.69	W=593 p-value=1.2e-6	yes
M. Eval.= 393000	*No Churn*	*34.82±0.66*	-	-

Cuadro 1.12: Wilcoxon test comparing the best fitness distribution of the EvAg model under different *degradation* rates. Results are obtained over 50 independent runs.

1.6.4. Summary

In this test-case, we have analysed the fault tolerance of the EvAg model when running on a computing platform that degrades following the modellization of the churn dynamics established by Stutzbach and Rejaie in [SR06]. To that aim, experiments were conducted on several instances of 2, 3 and 4-trap functions for different degradation rates.

Through the experimental results we can conclude that *churn* does not damage the scalability order of the algorithm and a small increase on the initial population size is enough to keep a reliable convergence towards optimal solutions. In that sense, large instances of trap functions have been tackled with success in spite of aggressive *churn* conditions in which peers depart from the system until 90 % of the initial configuration is left.

In addition, the approach shows to be resilient to *churn* with respect to execution time; once that the estimated population size guarantees a reliable convergence to the problem optimum, the departure of nodes does not inflict a penalisation on the computational effort.

Therefore, the EvAg model is fault-tolerant and implements a *graceful degradation* without any other extra mechanism than the emergent behaviour of the approach itself.

1.7. Conclusions

In this chapter, we have analysed the performance of the EvAg model in a simulated P2P environment following a experimental methodology based on the correct sizing of the populations. Experiments are conducted on trap functions with the aim of investigating the scalability of population sizes with increasing problem size and difficulty. In addition, in order to prove that the EvAg approach is *scalable* and *fault-tolerant*.

of the three test-cases:

The population size scales with potential order with respect to the problem size which demands for a big amount of resources.

model outperforms the scalability of canonical GAs, specially when

Given that traps have been designed to be difficult for EAs, these results should be easily extended to more general discrete or combinatorial opitmisation problems.

Bibliografía

[Ack87] David H. Ackley. *A connectionist machine for genetic hillclimbing*. Kluwer Academic Publishers, Norwell, MA, USA, 1987.

[AD08] Enrique Alba and Bernabe Dorronsoro. *Cellular Genetic Algorithms*, volume 42 of *Operations Research Computer Science Interfaces Series*. Springer, 2008.

[DAGT04a] B. Dorronsoro, E. Alba, M. Giacobini, and M. Tomassini. The influence of grid shape and asynchronicity on cellular evolutionary algorithms. In Y. Shi, editor, *IEEE International Conference on Evolutionary Computation*, pages 2152–2158, Portland, Oregon, June 20–23 2004. IEEE Press.

[DAGT04b] Bernabé Dorronsoro, Enrique Alba, Mario Giacobini, and Marco Tomassini. The influence of grid shape and asynchronicity on cellular evolutionary algorithms. In *In IEEE Congress on Evolutionary Computation (CEC2004*, pages 2152–2158, 2004.

[DG91] K. Deb and D.E. Goldberg. Analyzing deception in trap functions. In *Foundations of Genetic Algorithms, Morgan Kaufmann*, pages 93–108. Morgan Kaufmann, 1991.

[DHJ+07] Giuseppe DeCandia, Deniz Hastorun, Madan Jampani, Gunavardhan Kakulapati, Avinash Lakshman, Alex Pilchin, Swaminathan Sivasubramanian, Peter Vosshall, and Werner Vogels. Dynamo: amazon's highly available key-value store. *SIGOPS Oper. Syst. Rev.*, 41(6):205–220, 2007.

[ES93] L.J. Eshelman and J.D. Schaffer. Real-coded genetic algorithms and interval-schemata. In *Foundations of Genetic Algorithms 2*, pages 187–202. Morgan Kaufmann, San Mateo, CA, 1993.

[ES03] Agoston E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing*. SpringerVerlag, 2003.

[Fog94] L.J. Fogel. Evolutionary programming in perspective: the top-down view. In J.M. Zurada, R.J. Marks, and C. Robinson, editors, *Computational Intelligence: Imitating Life*. IEEE Press, Piscataway, NJ, 1994.

[GATT04] Mario Giacobini, Enrique Alba, Andrea Tettamanzi, and Marco Tomassini. Modeling selection intensity for toroidal cellular evolutionary algorithms. In *GECCO '04: Proceedings of the 2004 conference on Genetic and evolutionary computation*, LNCS, pages 1138–1149. Springer Berlin / Heidelberg, 2004.

[GDF01] The Gnutella Developer Forum GDF. The annotated gnutella protocol specification v0.4, 2001.

[Gol02] D.E. Goldberg. *The Design of Innovation - Lessons from and for Competent Genetic Algorithms*. Kluwer Academic Publishers, Norwell, MA, 2002.

[GPT06] Mario Giacobini, Mike Preuss, and Marco Tomassini. Effects of scale-free and small-world topologies on binary coded self-adaptive CEA. In Jens Gottlieb and Günther R. Raidl, editors, *Evolutionary Computation in Combinatorial Optimization – EvoCOP 2006*, volume 3906 of *LNCS*, pages 85–96, Budapest, 10-12 April 2006. Springer Verlag.

[GS89] Martina Gorges-Schleuter. Asparagos an asynchronous parallel genetic optimization strategy. In *Proceedings of the third international conference on Genetic algorithms*, pages 422–427, San Francisco, CA, USA, 1989. Morgan Kaufmann Publishers Inc.

[GTTA05] M. Giacobini, M. Tomassini, A. Tettamanzi, and E. Alba. Selection intensity in cellular evolutionary algorithms for regular lattices. *IEEE Transactions on Evolutionary Computation*, 9(5):489–505, 2005.

[HCPGM99] G. Harik, E. Cantú-Paz, D.E. Goldberg, and B. Miller. The gambler's ruin problem, genetic algorithms, and the sizing of populations. *Evolutionary Computation*, 7(3):231–253, 1999.

[JvS02] Márk Jelasity and Maarten van Steen. Large-scale newscast computing on the Internet. Technical Report IR-503, Vrije Universiteit Amsterdam, Department of Computer Science, Amsterdam, The Netherlands, October 2002.

[Koz92] J.R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA, 1992.

[LL07] Fernando G. Lobo and Claudio F. Lima. Adaptive population sizing schemes in genetic algorithms. In *Parameter Setting in Evolutionary Algorithms*, Studies in Computational Intelligence, pages 185–204. Springer Berlin / Heidelberg, 2007.

[RD01] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *In IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, pages 329–350, 2001.

[Rec94] I. Rechenberg. Evolution strategy. In J.M. Zurada, R.J. Marks, and C. Robinson, editors, *Computational Intelligence: Imitating Life*. IEEE Press, Piscataway, NJ, 1994.

[RFH+01] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Shenker. A scalable content addressable network. In *ACM SIGCOMM*, pages 161–172, 2001.

[Sas01] K. Sastry. Evaluation-relaxation schemes for genetic and evolutionary algorithms. Technical Report 2002004, University of Illinois at Urbana-Champaign, Urbana, IL., 2001.

[SdR99] Birgitt Schönfisch and André de Roos. Synchronous and asynchronous updating in cellular automata. *Biosystems*, 51(3):123 – 143, 1999.

[SMK+01] Ion Stoica, Robert Morris, David Karger, Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable Peer-To-Peer lookup service for internet applications. In *Proceedings of the 2001 ACM SIGCOMM Conference*, pages 149–160, 2001.

[SR06] Daniel Stutzbach and Reza Rejaie. Understanding churn in peer-to-peer networks. In *Proceedings of the 6th ACM SIGCOMM on Internet measurement (IMC 2006)*, pages 189–202, New York, NY, USA, 2006. ACM Press.

[Thi99] Dirk Thierens. Scalability problems of simple genetic algorithms. *Evolutionary Computation*, 7(4):331–352, 1999.

[Tom05] Marco Tomassini. *Spatially Structured Evolutionary Algorithms: Artificial Evolution in Space and Time (Natural Computing Series)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.

[ZKJ01] B. Y. Zhao, J. D. Kubiatowicz, and A. D. Joseph. Tapestry: An infrastructure for fault-tolerant wide-area location and routing. Technical Report UCB/CSD-01-1141, UC Berkeley, April 2001.