Fault Tolerant and Dynamic Evolutionary Optimization Engines

Alicia Morales Reyes

A thesis submitted for the degree of Doctor of Philosophy

The University of Edinburgh

September 2010

Declaration of originality

I hereby declare that the research recorded in this thesis and the thesis itself was composed and originated entirely by myself in the School of Engineering at the University of Edinburgh.

Alicia Morales Reyes, September 2010.

Acknowledgements

This research has been supported by the Mexican people through the Mexican National Council for Science and Technology (CONACyT). Studentship 168016.

I would like to acknowledge the contribution of Prof. Tughrul Arslan and Dr. Ahmet T. Erdogan towards this research.

I would like to thank Prof. Xin Yao who acted as external examiner and Dr. Alister Hamilton who acted as internal examiner for this dissertation.

I would like to dedicate this thesis to my family for their unfettered support and encouragement.

Abstract

Mimicking natural evolution to solve hard optimization problems has played an important role in the artificial intelligence arena. Such techniques are broadly classified as Evolutionary Algorithms (EAs) and have been investigated for around four decades during which important contributions and advances have been made.

One main evolutionary technique which has been widely investigated is the Genetic Algorithm (GA). GAs are stochastic search techniques that follow the Darwinian principle of evolution. Their application in the solution of hard optimization problems has been very successful. Indeed multi-dimensional problems presenting difficult search spaces with characteristics such as multi-modality, epistasis, non regularity, deceptiveness, etc., have all been effectively tackled by GAs.

In this research, a competitive form of GAs known as fine or cellular GAs (cGAs) are investigated, because of their suitability for System on Chip (SoC) implementation when tackling real-time problems. Cellular GAs have also attracted the attention of researchers due to their high performance, ease of implementation and massive parallelism. In addition, cGAs inherently possess a number of structural configuration parameters which make them capable of sustaining diversity during evolution and therefore of promoting an adequate balance between exploitative and explorative stages of the search.

The fast technological development of Integrated Circuits (ICs) has allowed a considerable increase in compactness and therefore in density. As a result, it is nowadays possible to have millions of gates and transistor based circuits in very small silicon areas. Operational complexity has also significantly increased and consequently other setbacks have emerged, such as the presence of faults that commonly appear in the form of single or multiple bit flips. Tough environmental or time dependent operating conditions can trigger faults in registers and memory allocations due to induced radia-

tion, electron migration and dielectric breakdown. These kinds of faults are known as Single Event Effects (SEEs).

Research has shown that an effective way of dealing with SEEs consists of a combination of hardware and software mitigation techniques to overcome faulty scenarios. Permanent faults known as Single Hard Errors (SHEs) and temporary faults known as Single Event Upsets (SEUs) are common SEEs. This thesis aims to investigate the inherent abilities of cellular GAs to deal with SHEs and SEUs at algorithmic level. A hard real-time application is targeted: calculating the attitude parameters for navigation in vehicles using Global Positioning System (GPS) technology. Faulty critical data, which can cause a system's functionality to fail, are evaluated. The proposed mitigation techniques show cGAs ability to deal with up to 40% stuck at zero and 30% stuck at one faults in chromosomes bits and fitness score cells.

Due to the non-deterministic nature of GAs, dynamic on-the-fly algorithmic and parametric configuration has also attracted the attention of researchers. In this respect, the structural properties of cellular GAs provide a valuable attribute to influence their selection pressure. This helps to maintain an adequate exploitation-exploration trade-off, either from a pure topological perspective or through genetic operations that also make use of structural characteristics in cGAs. These properties, unique to cGAs, are further investigated in this thesis through a set of middle to high difficulty benchmark problems. Experimental results show that the proposed dynamic techniques enhance the overall performance of cGAs in most benchmark problems.

Finally, being structurally attached, the dimensionality of cellular GAs is another line of investigation. 1D and 2D structures have normally been used to test cGAs at algorithm and implementation levels. Although 3D-cGAs are an immediate extension, not enough attention has been paid to them, and so a comparative study on the dimensionality of cGAs is carried out. Having shorter radii, 3D-cGAs present a faster dissemination of solutions and have denser neighbourhoods. Empirical results reported in this thesis show that 3D-cGAs achieve better efficiency when solving multi-modal and epistatic problems. In future, the performance improvements of 3D-cGAs will merge with the latest benefits that 3D integration technology has demonstrated, such as reductions in routing length, in interconnection delays and in power consumption.

Contents

List of Figures

List of Tables

Acronyms and Abbreviations

2D	Two Dimensions
3D	Three Dimensions
AFM	Ambiguity Function Method
ANOVA	Analysis of Variance
BT	Binary Tournament
CA	Cellular Automata
CDA	Constant Dynamic Anisotropic selection
cGA	Cellular Genetic Algorithm
CLR	Constant Lattice Reconfiguration
dGA	Distributed Genetic Algorithm
EA	Evolutionary Algorithm
EHW	Evolvable Hardware
FIT	Faults in Time
FMS	Frequency Modulation Sound
GA	Genetic Algorithm
GDA	Genotypic Dynamic Anisotropic selection
genGA	Generational Genetic Algorithm
GLR	Genotypic Lattice Reconfiguration
GPS	Global Positioning System
IC	Integrated Circuits
MBU	Multiple Bit Upset
MIMD	Multiple Instruction Multiple Data
MMDP	Massively Multi-Modal Deceptive Problem
MTTP	Minimum Tardy Task Problem
NGR	Neighbourhood to Grid Ratio

NP	Non-deterministic Polynomial time
PDA	Phenotypic Dynamic Anisotropic selection
PE	Processor Element
PGA	Parallel Genetic Algorithm
PLR	Phenotypic Lattice Reconfiguration
RW	Roulette-Wheel
SEE	Single Event Effect
SER	Soft Error Rate
SEU	Single Event Upset
SHE	Single Hard Error
SLE	System of Linear Equations
SoC	System on Chip
ssGA	Steady State Genetic Algorithm
VLS	Variable Neighbourhood Search
VLSI	Very-Large-Scale Integration

Chapter 1

Introduction

1.1 Motivation

Optimization has been an essential part of systems design and implementation. Among classical optimization techniques are the theory of minima and maxima and the calculus of variations. At the time when the field of optimization reached a certain level of maturity (in the 1960's) and researchers were very much attracted to the area, biologically inspired techniques started to appear [1]. Thus, Evolutionary Algorithms (EAs) are computational techniques of the artificial intelligence arena that follow the Darwinian principle of evolution. Among those, Genetic Algorithms (GAs) stand out as a powerful tool with which to solve difficult optimization problems in a wide variety of application scopes. GAs were introduced in 1975 by John Holland and since then researchers have dedicated extensive investigations to their improvement and to making them available to tackle real world problems.

GAs possess a simple processing structure that works on an initial randomly generated population. Individuals (chromosomes) are strings normally represented through binary, grey or real encoding. An iterative process of selection, recombination, mutation and replacement among individuals takes place. Each iteration or each sequence of genetic operations is known as a generation. In each generation a new population is generated, either with newly created individuals or with a mixture of individuals from the previous generation and new individuals. The evolutionary process stops either when a pre-defined number of generations is reached or the solution is found [2].

Each individual independently encodes a solution to the problem and its success depends on its own and the selected partner's quality for reproduction; thus GAs can easily be parallelized. However, in order to do that and from an implementation perspective, the population's grain should be defined. Coarse grain GAs are those in which the population is divided and parallelized while panmictic interaction (all individuals in the population can potentially mate any other individual) is maintained inside each sub-population. In this way, parallelism is carried out at sub-population level. On the other hand, fine grain or cellular GAs (cGAs) define a population topology, normally by using a toroidal grid structure, placing one individual per grid position that interacts only through nearby neighbours [3].

One of the main issues in evolutionary algorithms is the preservation of diversity. Excessive exploitation could lead the search to stagnate or to converge to a local optimum. In contrast, too much exploration could significantly increase the number of evaluations required to find the solution, or in the worst case scenario to fail or to converge to a local optima. Maintaining a balance between exploitation and exploration is a major challenge in evolutionary techniques. In many cases, this balance is pursued through changes in the genetic operations of the main reproductive cycle [4].

In the fault tolerant arena, diversity could also be affected by the presence of faults known as Single Event Effects (SEEs) which are logical temporary or permanent faults that could affect main data structures in evolutionary optimization engines, such as the phenotypes or genotypes registers. In this thesis the inherent properties of cGAs are investigated as mitigation techniques to sustain diversity while dealing with faults at algorithmic level. Those characteristics include the size and shape of the local neighbourhood, the implicit migration due to the smooth spread of solutions as a result of neighbourhoods overlapping, or the explicit migration with policies that allow individuals to avoid faulty neighbourhoods as an attempt at survival aiming to converge to the global optimum.

Genetic Algorithms possess significant attributes for dynamic adaptation while searching for the solution to a problem. A major challenge in the EAs arena is to take advantage of those attributes in order to improve the algorithmic performance without increasing the computational cost. Dynamic parameter setting of genetic operations is an area of research that has been widely investigated since the very beginning of EAs. Perhaps less visible are other properties of certain evolutionary techniques such as those that attracted the Author's attention: the structural properties of cellular GAs and their effect on the evolutionary process.

1.2 Goals

The aim of this thesis is to propose fault tolerant and dynamic algorithmic techniques to further investigate the ability of fine grained Genetic Algorithms to improve their performance when tackling difficult optimization problems. The work presented in this thesis is organized into three main parts; the development of mitigation techniques that take advantage of the inherent properties of cellular GAs to overcome faulty scenarios; the development of dynamic criteria to improve the performance of cellular GAs while taking advantage of their structural properties, and thirdly, the investigation of the implication of dimensionality in cellular GAs.

The research in this thesis started as a continuation of ongoing research at the System Level Integration research group. The investigation explores the fault tolerant arena to deal with two kinds of Single Event Effects (SEEs): Single Hard Errors (SHEs), permanent faults, and Single Event Upsets (SEUs), temporary faults, affecting main data structures in a cellular GA based reconfigurable architecture. The targeted application is the GPS attitude determination problem, a hard real-time constrained problem used for vehicles' navigation. An interesting area of research is the development of mitigation techniques to deal with faults caused, among other factors, by radiation affecting a system operating in harsh environmental conditions. The presence of SEEs in main system data such as phenotypes or genotypes registers and memory allocations, would cause the searching process of a cGA to fail. In order to deal with failures without significantly increasing the use of physical or computational resources, properties specific to cellular GAs are investigated in this thesis as a resource for mitigation of SEEs.

This thesis investigates fine grained GAs while tackling difficult optimization problems from a particular perspective: that is the way in which the inherent structural and processing properties of cellular GAs are advantageous as an approach for diversity preservation. Dynamic on-the-fly modification of EAs parameters is an important arena. In this respect, cellular GAs provide a distinct feature that could be added to the investigation of dynamics in EAs. The principal characteristic of cellular GAs is the use of a decentralized and structured population together with a neighbourhood that locally delimits the interaction of individuals, providing, at the same time, a double approach to deal with the search space. Locally, solutions are exploited, while globally throughout the grid, the landscape is explored. The basis of the second part of this study is to further investigate the effect of dynamically changing the topology configuration at run time, in order to contribute to sustaining diversity by appropriately balancing the exploitation-exploration trade-off.

The third part of this investigation aims at providing a better understanding of the dimensionality of cGAs and its effect on their performance. Cellular GAs are normally implemented on 1D or 2D grid topologies, toroidally connected. 3D-cGAs present characteristics such as shorter radii that allow a faster spread of solutions; a larger number of individuals in the neighbourhood that locally permits more diversity of solutions for selection, amongst others. An empirical study is carried out in order to determine whether 3D-cGAs achieve better performances than 2D-cGAs without increasing the number of individuals in the population. If the outcome is positive, in future combining 3D-cGAs at algorithmic level with the advances on 3D-IC technology will represent a robust platform for high performance optimization engines.

1.3 Contribution to knowledge

This study aims at providing a deeper understanding of the abilities cellular GAs possess from a structural point-of-view. The need of a structure and consequently the configuration of the population topology and the local neighbourhood are aspects that differentiate cGAs from panmictic and other parallel GAs approaches. This study concentrates on demonstrating that it is possible to take advantage of those structural characteristics to extend the application of cGAs to the fault tolerant arena and to influence the exploration-exploitation trade-off, a main challenge in evolutionary techniques. The thesis presents the following novel ideas to improve cGAs performance:

- Migration operation as a mitigation technique to overcome SHEs and implicitly SEUs affecting fitness score registers. Several migration policies were defined and evaluated based on random, best neighbourhoods and best neighbours selection.
- Configuration of the local neighbourhood as a way of modifying the induced selective pressure and to provide every individual with more alternatives for selection and therefore to avoid faulty individuals in the phenotypic space.
- Modification of the topology shape during evolution to provide a lower selective

pressure and therefore a more explorative search together with an explicit migration mechanism that allows non-faulty individuals to avoid faulty ones in the genotypic space.

- Improvement in results accuracy by using a distributed parallel approach that provides two levels of selective pressure.
- Application of previous distributed parallel approach to balance the exploration-exploitation trade-off as a way of dealing with the loss of diversity due to SHEs or SEUs affecting the phenotypic space.
- Dynamic internal reconfiguration of the population's topology while maintaining the adjacency of individuals without inducing an explicit migration mechanism.
- Dynamic anisotropic selection which makes use of individuals location inside the neighbourhood an dynamically assigns probabilities for selection.
- Dimensionality as a way to improve cGAs performance as a result of a faster spread of solutions due to shorter radii and having locally more alternatives for reproduction.

1.4 Thesis Organization

The remainder of this thesis is divided into five chapters and is organized as follows:

- **Chapter 2** gives an introduction to Evolutionary Algorithms with a particular emphasis on Genetic Algorithms. A review of parallel GAs approaches is also provided in order to situate fine grained or cellular GAs, which are the main subject of this research. A detailed review of cellular GAs is also presented including an initial empirical comparison with standard GAs. This chapter also includes an overview of several standard and cellular GA based reconfigurable architectures. Finally, the topic of fault tolerance is presented in order to introduce the fault tolerant perspective examined in Chapter 3.
- **Chapter 3** presents a study on fault tolerance while tackling a hard real-time constrained problem: the GPS (Global Positioning System) attitude determination problem. Determining the attitude parameters of a vehicle using GPS technology represents a system's simplification and a significant cost reduction in

comparison to traditional systems for vehicle navigation, such as inertial navigation systems [5]. As a result, a fault tolerant platform is proposed based on the inherent ability of cGAs to deal with Single Hard Errors (SHE) and Single Event Upsets (SEUs) which could permanently or temporarily affect its operation. Two faulty scenarios are evaluated: SHEs or SEUs affecting phenotypes or genotypes registers. Moreover, a distributed approach is also investigated as a fault tolerant platform using migration as a mitigation technique to overcome faulty scenarios.

- **Chapter 4** assesses an extended set of difficult optimization problems to further investigate the effect of cGAs structural properties on their performance. Continuous mathematical functions, such as the Rastrigin, Griewank and Langerman functions, are explored on the discrete domain using binary representation. Real problems such as the Frequency Modulation Sound (FMS) problem, the GPS attitude determination problem and the System of Linear Equations problem are also tackled. Finally, combinatorial problems such as the Massively Multi-modal Deceptive Problem (MMDP), the Minimum Tardy Task Problem (MTTP) and the P-Peaks problem are approached through several dynamic criteria.

  The main objective in the second part of this research is to maintain the canonical form of cGAs while taking advantage of their structural properties. The aim is to avoid increasing their computational cost which would also imply an increase in the amount of resources required for their implementation. Two experimental set ups are proposed: 1) a gradual inclusion of internally different lattice topologies is evaluated. 2) three experimental cases proposed by Dorronsoro et al. in [6], including static, pre-programmed and adaptive reconfiguration mechanisms, are directly compared to the internal constant and adaptive lattice reconfiguration techniques proposed here.

- **Chapter 5** is dedicated to investigating and comparing 2D versus 3D cellular GAs. Increasing dimensionality in cellular GAs provokes different selection pressures on the same population sizes. This chapter aims to research the advantages in the performance of cGAs when using 3D toroidal structures for the development of future 3D optimization engines. This chapter is also the starting point for current ongoing research in the System Level Integration research group.

- **Chapter 6** presents a summary and the conclusions of this thesis including the main contributions of this research. Finally, some guidelines for future research are presented.

Chapter 2

Literature Review

In this chapter, a general overview of Evolutionary Algorithms (EAs) is presented, together with a comprehensive review of cellular Genetic Algorithms (cGAs). Being part of a wide algorithmic family such as EAs, cGAs have demonstrated to be a competitive tool showing interesting abilities to deal with hard optimization problems, outperforming in several cases the traditional standard GAs and other well known parallel GAs approaches.

2.1 Evolutionary Algorithms

Evolutionary Algorithms are non-deterministic techniques that follow the Darwinian Principle of Evolution. A population is randomly or under certain conditions created and formed by individuals. Normally, individuals are encoded using binary, grey or real encoding [7]. Competition for survival among individuals determines which ones will reproduce and pass their genetic material to new individuals in following generations.

There is a set of basic genetic operations that EAs use during evolution [8]. Individuals are picked out in pairs following a predefined criterion (selection), pairs of individuals are mated, this operation is commonly known as crossover (recombination). Then, the offspring alleles can be randomly modified in order to recover genetic diversity or to be able to avoid convergence to local optima (mutation). A random genetic change can determinately modify the search process. Thus, careful attention should be paid to this operation. Finally, the offspring are introduced into the population following certain rules, as an example, they can replace old individuals in the current population or create a new population composed by old and new individuals.

EAs are classified in four groups: Genetic Algorithms (GAs), Genetic Programming (GP), Evolutionary Strategies (ES) and Evolutionary Programming (EP). Each evolutionary technique works in specific application arenas but commonly apply, with lower or higher priority and strength, the following genetic operations: selection, recombination and mutation. For example, GP operates on chromosomes of variable length, while GAs maintain constant chromosomes length. Therefore, GP needs to use modified genetic operators [4]. In cases such as GP and ES, there is no difference between genotypic and phenotypic representation, while GAs clearly distinguish between phenotype and genotype spaces. These techniques are relatively new, GAs were proposed by John Holland in 1975, John Koza proposed GP in 1992, while in 1960 ES and EP were created by Bienert, Rechenber and Schwefel and Lawrence Fogel respectively [9].

Evolutionary techniques have the disadvantage of providing, in some cases, less accurate solutions in comparison to an application specific algorithm. However, EAs offer important advantages, such as their inherent parallelism. Moreover, due to the fast technological development and the processing power nowadays available, EAs parallelism has been exploited tackling difficult and real problems. This research focuses on the study of cellular Genetic Algorithms, a subgroup of Genetic Algorithms that works with decentralized populations. In the next section an overview of Genetic Algorithms is presented.

2.2 Genetic Algorithms

Genetic Algorithms are maybe the most widely studied evolutionary technique. As mentioned above, GAs were proposed by Holland in the seventies. He proposed the use of this algorithmic approach to solve practical problems rather than for simulating biological systems [10]. Successfully, his idea was rapidly accepted and spread. However, Holland's aim was not only to solve application specific problems but to create a general framework for a kind of adaptive system, in which individuals represent encoded programs that communicate through binary sensors.

As search strategies, GAs aim at obtaining the best approximate solution for a given problem. Theoretically, GAs define two spaces: the representation or genotypic space in which solutions are encoded into chromosomes, normally using binary strings but other ad-hoc representations are also used. And the search or phenotypic space, where chromosomes are translated into actual solutions. Mapping between genotype and phenotype spaces is done by an encoding function. After that, phenotypes are evaluated by the fitness function. Once individuals fitness scores are calculated, parents selection is performed. After selection, pairs of individuals are recombined and mutation is then applied to the offspring. The final step in the reproductive cycle is the insertion of the new individuals. Hence, replacement policies are defined, for example, new individuals can replace current ones always or only if offspring provide better solutions [3].

In Algorithm 1, a standard GA pseudocode is presented. The basic reproductive cycle consists of three main operations: selection, recombination and mutation, that can be tuned accordingly to problem specific characteristics [2].

Algorithm 1 Genetic Algorithm Pseudocode

```
 1: procedure GA
 2:     $(x) \leftarrow random\,(x_0)$                                ▷ initial population
 3:     $(f) \leftarrow evaluation\,(x)$                             ▷ evaluation
 4:     while $k \leftarrow 1, generations,$   or   $\bar{f} <= threshold$ do
 5:         $(f', x') \leftarrow selection\,(f, x)$                  ▷ parents selection
 6:         $(x'') \leftarrow recombination\,(x')$                   ▷ parents recombination
 7:         $(x''') \leftarrow mutation\,(x'')$                      ▷ offspring mutation
 8:         $(f_{new}) \leftarrow evaluation\,(x''')$                ▷ offspring evaluation
 9:         $(f, x) \leftarrow replacement\,(f_{new}, x_{new})$      ▷ individuals replacement
10:     end while
11: end procedure
```

There are several selection methods, among those commonly used are: tournament selection, roulette wheel selection, deterministic selection, among others. Tournament selection consists in choosing a specific number of individuals on a random base. These individuals are pairs of parents that will reproduce. Individuals selection can be performed with or without replacement. The size of the tournament would vary based on the experimental constraints. In this thesis, binary tournament (BT) selection is used on the experimental set ups. BT randomly selects two individuals who compete for a parent place. In standard or panmictic GAs, a mating pool is created by selected individuals that would mate according to recombination and mutation operations.

Another commonly used method is the roulette-wheel selection (RW). It assigns, as in a roulette-wheel, slots for each individual. The size of the slot depends on its fitness.

Individuals with higher fitness scores have wider slots, and therefore higher probability for selection. Instead, individuals with lower fitness scores would have narrower slots in the wheel. Individuals can then be selected with or without replacement, in a similar way to tournament selection.

Deterministic selection is less flexible. It consist in defining a fixed criterion for individuals selection. For example, only individuals with highest fitness scores are considered for selection. Depending on the problem, this method induces more rapidly the loss of diversity. Systematically choosing only highly fitted individuals would probably stagnate the evolutionary process, as there is not diversity among individuals to explore or exploit diverse areas of the search space.

Recombination is an operation closely related to individuals' representation. For binary chromosomes representation, single or double point crossover is commonly applied. For real or float chromosomes representation, arithmetic crossover is implemented [7, 11]. Mutation, which aims at introducing genetic diversity in each reproductive cycle, is also an operator that needs to be finely tuned. In fact, mutation by itself started a subclass of evolutionary techniques known as Evolutionary Strategies as their primary operator. Finally, replacement policies are also fundamental in GAs. In fact, two improved versions known as steady state and generational GAs directly depend on replacement policies. In the next section, an overview of parallel Genetic Algorithms is presented.

2.3 Parallel Genetic Algorithms

Evolutionary techniques are prone to parallelization. In particular, GAs have been widely investigated in this regard and parallel approaches have emerged since the very early years GAs came up. Parallelization in GAs can be carried out in several ways at different algorithmic and implementation levels, aiming at improving their efficiency and efficacy when tackling difficult optimization problems.

A rough classification for Parallel Genetic Algorithms (PGAs) groups them by grain size in coarse and fine PGAs. Coarse PGAs mainly consist in dividing a whole panmictic population into several sub-populations with individuals migrating among them. Thus, reducing the number of individuals each processing unit should deal with. A consequent link between algorithm and implementation is thus appreciated. However,

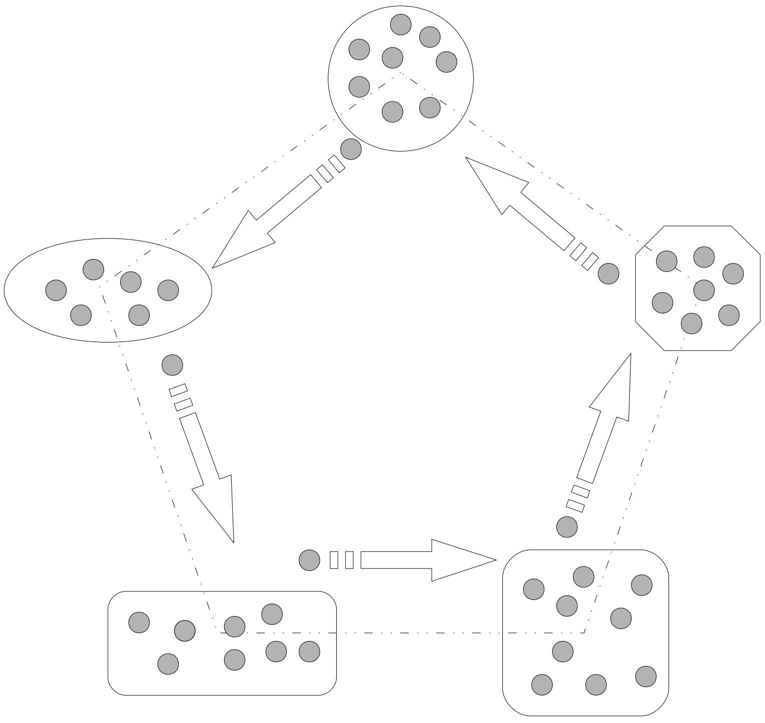


Figure 2.1: A coarse or distributed Genetic Algorithm

implementations using a single processing unit to deal with several sub-populations have also been developed and improved performances have been achieved [12]. Coarse PGAs, also named as distributed GAs (dGAs), gave place to new research arenas in the field of GAs. Having different parameters configuration to perform genetic operations and/or implementing different grains, and/or having distinct chromosome representation in each sub-population, were among others new possibilities to explore that would possibly lead to improve GAs performance [13, 14, 15, 16]. In Figure 2.1 a dGA diagram is shown. Each sub-population is centralized and migration policies need to be defined by rate and frequency. It is also necessary to define if migration would be synchronous or asynchronous. In [17] an interesting approach of heterogeneous distributed PGA was presented. Using a hypercube structure with each vertex holding one sub-population. All sub-populations located on one side of the cube perform more exploitative or more explorative search. Internally, each sub-population implements distinct configurations for the genetic operations in order to promote different exploitation-exploration levels. Hence, diversity is well maintained globally while locally the accuracy of results is improved.

The main difference of fine grained PGAs with respect to coarse or distributed

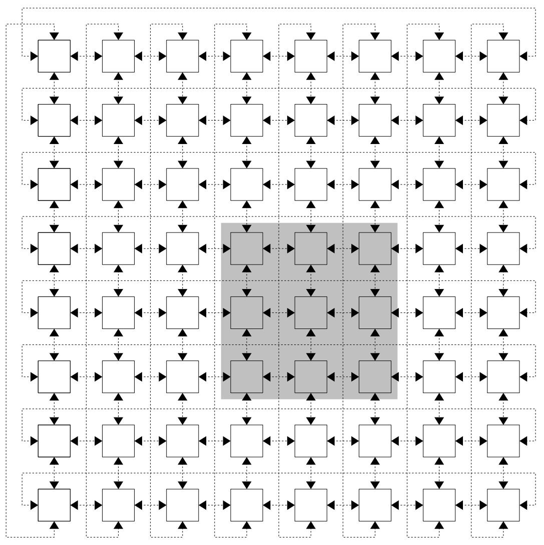


Figure 2.2: A fine or cellular Genetic Algorithm

PGAs is the decentralization of the population and the consequent also decentralized selection of individuals. Fine PGAs are also known as cellular GAs (cGAs), in Figure 2.2, a common square population topology following a torus like grid with wraparound edges is drawn. A single individual is located per grid position. Around each individual, a neighbourhood is defined, among its individuals is the second parent that would be selected for reproduction. In Figure 2.2 a local Moore neighbourhood with 9 individuals is drawn as an example.

In cellular GAs, every individual interacts with its close neighbours, providing a smooth diffusion of solutions throughout the grid due to the neighbourhoods overlapping [18]. In coarse PGAs, individuals association is loose while it is strong in cellular GAs. These among other characteristics specific to cGAs makes them a powerful tool to tackle, from a variety of perspectives, difficult search spaces of hard problems.

Researchers have pursued several goals through the parallelization of GAs, such as finding new routes to solve a problem, parallel searching of different regions in the search space, better convergence and processing times in comparison to sequential versions, simplified parallelization techniques, among others [19]. Parallel GAs allow several levels for parallelization, and combining panmictic and cellular sub-populations in dGAs is an interesting approach detailed in Subsection 2.3.1.

An important operation in evolutionary processes and particularly in dGAs is the migration of individuals among sub-populations. Once individuals are evolving in each sub-population, one or more of them should be selected to migrate and replace individuals in other sub-populations. In this regard, migration policies should be defined not only in terms of frequency and rate, but also policies for selection of migrants and replacement of individuals after arriving to new sub-populations are required [20]. Moreover, migration among sub-population is performed synchronously or asynchronously. Synchronous migration expect all sub-populations to be at the same state and then individuals migrate at the same time. On the other hand, asynchronous migration allow each sub-population to determine when individuals should migrate. In Subsection 2.4.3 a discussion on migration synchronism is presented.

Cantú-Paz et al. provided a complete study on dGAs, from a theoretical and practical perspective [3]. Initially, a proposal to solve the problem of determining an adequate population size to achieve certain solution quality is analysed. A mathematical model using the building blocks hypothesis, proposed by Holland and continued by Goldberg [2], is tested on several problems with accurate results that also showed good scalability.

To find bounding cases for dGAs, in terms of efficiency, the interaction among sub-populations and its effect on the quality of solutions was also analysed in [3]. It has been previously investigated that the quality of solutions changes according to the way sub-populations interact. Having loosely to null communication among sub-populations results in less accurate solutions while better ones are obtained through panmictic approaches. However, when individuals migrate among sub-populations, at low or middle frequency, the overall dGAs performance improves, achieving in several cases super linear speedups and better quality of solutions. Due to its essential role in dGAs, migration is analysed in detail in Subsection 2.3.2 and in Subsection 2.4.3.1 migration synchronism is explained.

2.3.1 Homogeneous and Heterogeneous PGAs

Different scenarios to define homogeneous or heterogeneous distributed PGAs exist. Roughly, at algorithmic level the same or different configuration for the genetic operations can be applied in each sub-population. From these genetic operations, distinct configuration parameters can tune the search on each sub-population. For example, different crossover definitions or mutation probabilities operate individuals in each sub-population. Also, individuals representation can be different among sub-populations. At a higher level, only panmictic GAs or cellular GAs or a mixture of both can be implemented. On the other hand, at implementation level, heterogeneous computing has been used in the study of PGAs. In [21] different processing platforms were available to deal with PGAs sub-populations.

In [12], several continuous functions were tackled through an improved version of an heterogeneous PGA platform using real chromosomes encoding. A double hypercubic topology was compared with a single hypercubic topology. Sub-populations located at the vertexes on each face were configured to promote a more explorative or more exploitative search through genetic operations tuning. Recombination provides a more explorative behaviour while mutation encourages solutions exploitation. Migration between sub-populations allows the search to be refined among explorative and exploitative levels performed by each sub-population. Results showed the importance of asynchronism in migration and the direct effect on execution time as well as in the average number of evaluations for both topologies. In terms of the hit rate, a double hypercubic topology outperformed a single one.

At implementation level, heterogeneous computing has also been used to assess PGAs. Alba et al. proposed a heterogeneous platform in which different operating systems with distinct security restrictions execute a distributed PGA model [21]. In an unidirectional ring, each sub-population executes a steady state GA. In every generation one offspring is produced and replaces the worst individual. Asynchronous migration between sub-populations is applied due to its proved better performance. Two combinatorial problems were tackled to evaluate the heterogeneous PGA performance. Comparisons were carried out against a homogeneous processing platform. Improved results in terms of the number of evaluations were achieved by the heterogeneous platform due to the exploitation of solutions at different rates.

2.3.2 Migration in PGAs

Migration in PGAs can be analysed from multiple perspectives. In coarse PGAs, where panmictic or fine sub-populations evolve in parallel, migration should be considered as a mechanism to introduce new individuals and therefore to promote or restore diversity among sub-populations [22]. In contrast, an implicit mechanism for migration in fine or cellular PGAs occurs on a single decentralized population, due to the neighbourhoods overlapping, where individuals slowly spread their solution throughout the array. This is a migration mechanism inherent to cellular GAs processing structure. It does not need any frequency or rate for migration, or any selection and replacement policies to be defined.

In this section, an overview of migration policies used in coarse PGAs is carried out. Although these policies are more appropriate for coarse PGAs, an added level for migration can also be implemented on fine PGAs, apart from the natural migration mechanism described before.

Cantu-Paz dedicated a complete study on PGAs emphasizing the importance of migration rules and mechanisms [3]. The aim of his study was to stablish appropriate criteria for migration which would improve PGAs performance. It was suggested that maintaining certain isolation level among sub-populations is a necessary condition. If a sub-population remains in complete isolation, improvements in its diversity are minimal and therefore its ability to solve the problem is reduced. Yet, if permanent communication among sub-populations is considered, the cost increases dramatically and becomes impractical. Thus, the aim of migration is to avoid the total isolation of sub-populations while maintaining an acceptable level of communication among them.

In [23] a theoretical study to calculate the appropriate number of processors needed to minimize the execution time in PGAs is carried out. An experimental set up is implemented considering bounding cases in terms of connectivity among sub-populations. That is the case of having the maximum and minimum migration frequencies and rates. Theoretical conclusions for a single population indicated that an optimal number of processors is given by $O\left(\sqrt{\frac{nT_f}{T_c}}\right)$, where n is the population size, T_f is the time to calculate the fitness of one individual and T_c is the average time for communication of one processor. This means that dividing a single population into multiple processors reduces the overall execution time. Moreover, the same conclusion was achieved for a multi-population approach considering addition separable functions [24]. Therefore, using several processors while implementing PGAs has a positive impact in reducing the execution time.

Another study to analyse the effect of migration policies in coarse PGAs was presented in [20]. The main objective was to determine how the migration operation affects the induced selection pressure and to implicitly provide an explanation for the super-linear speedups achieved by PGAs. The way individuals are selected for migration and replacement has a considerable effect in selection pressure and therefore in PGAs convergence time. Selecting and replacing individuals by fitness score drastically increases or decreases the selection pressure. It was observed that higher migration rates reported also an increase in selection pressure. High selection pressures lead the search to a faster convergence, however the risk of premature convergence at a local optimum is increased.

Although migration has mainly been investigated considering coarse PGAs; there are also studies that have approach migration in fine PGAs, not only considering the inherent induced migration due to neighbourhoods overlapping but establishing some criteria to migrate individuals among lattice locations. In [25], a binary tree-like structure was proposed to divide a square grid topology in concentric formations of individuals, implementing in each two different migration policies. The exchange of individuals among formations defined by this structure slows down the search process, allowing more exploration. Results showed a better performance on constrained problems. Mutation is applied to avoid the presence of super individuals that could lead the search to stagnate while taking over the entire population much quicker. In the next section all details about fine or cellular GAs are provided, including examples not only at algorithmic but also at implementation level.

2.4 Cellular Genetic Algorithms

Cellular Genetic Algorithms are fine grained GAs consisting of a decentralized population where individuals interact with others located at nearby positions. Normally, cGAs are implemented on a n-dimensional toroidal grid with wraparound edges following traditional geometrical shapes, such as square, rectangular or linear structures.

Cellular GAs processing matches that of a Cellular Automata (CA). CAs ideally represent discrete physical systems in space and time. Determining a global rule that define a specific behaviour built from local rules is achieved by CAs. In [26, 27], an interesting analysis in this work considers the distance between a cell and its neighbours as a determinant factor to avoid any loss of new information produced due to cells interaction. Hence, a maximum distance between two cells and an average distance among cells are used as metrics to evaluate CAs performance. On one dimension CAs, the average distance among cells is larger than having two or more dimensions, where distance is reduced and more individuals are reached at a time.

In [28] one dimension CAs are extended to two and three dimensions. GAs were used in combination with CAs to evolve the desired global behaviour. Results showed that higher dimensions perform significantly better when evolving a global behaviour. This combination of a CA and a GA is in EAs classification known as a cellular GA, with the difference of having a more elaborated representation for each cell: chromosomes. In [26] observations were made as regards to cells or individuals communication reaching long distances, that are allowed by neighbourhoods interactions or overlapping. Thus, information travels among cells, combining and generating new information. Having more neighbours sharing information allows individuals to accelerate the process of creating new behaviours to fit the problem with a minimum loss of information. Three combinatorial problems were tackled, the majority problem, the checker board problem and the evolution of bitmaps. Results showed that multi dimensional CAs solve these problems much faster than one dimension CAs.

In Algorithm 2, the basic pseudocode for cellular GAs is presented. An initial random population is generated to fill all positions in the toroidal like grid, placing one individual per location. This initial population is evaluated and the main reproductive cycle starts. A local neighbourhood is defined by closely located individuals surrounding each individual. From the neighbourhood, both parents or only one parent is selected. Details about cGAs configuration are provided in the following sections. There are different selections methods that can be used in cGAs, those commonly implemented on standards GAs, such as binary tournament, roulette-wheel or deterministic selection. There are also other selection methods that only apply on decentralized GAs, such as anisotropic or centric selection [29, 30]. After selection, parents are recombined and offspring are mutated, there is no difference in the application of these operations to that used in standard GAs. Individuals' updating can be carried out synchronously or asynchronously. If individuals are updated synchronously, a temporary array would allocate the offspring until the entire population has fully reproduced and then the temporary population will replace the current population. The reproductive cycle is repeated until the stop condition is fulfilled. Subsection 2.4.3 is dedicated to explain synchronism in PGAs with particular attention to its role in cGAs. In the following section, the topic of selection pressure in cGAs is presented.

2.4.1 Selection Pressure due to Structural Properties

In this thesis, particular attention has been paid to cGAs structural properties and how these can benefit the searching performance of this evolutionary technique. The

Algorithm 2 Cellular Genetic Algorithm Pseudocode

```
1:  procedure CGA
2:      $(x) \leftarrow random\,(x_0)$                                        ▷ initial population
3:      $(f) \leftarrow evaluation\,(x)$                                      ▷ evaluation
4:      while $k \leftarrow 1, generations,$ or $\bar{f} <= threshold$ do
5:          for $i \leftarrow 1, populationSize$ do
6:              $(f_1, f_2) \leftarrow selection\,(f, f_N, f_E, f_S, f_W)$          ▷ parents selection
7:              $(x_1, x_2) \leftarrow selection\,(x, x_N, x_E, x_S, x_W)$          ▷ in L5 neighbourhood
8:              $(x'_1, x'_2) \leftarrow recombination\,(x'_1, x'_2)$            ▷ parents recombination
9:              $(x''_1, x''_2) \leftarrow mutation\,(x'_1, x'_2)$               ▷ offspring mutation
10:             $(f_{new}) \leftarrow evaluation\,(x''_1, x''_2)$              ▷ offspring evaluation
11:             $(f, x) \leftarrow replacement\,(f_{new}, x_{new})$            ▷ individuals replacement
12:         end for
13:     end while
14: end procedure
```

shape of the population topology, the number of individuals a neighbourhood has, the shape of the neighbourhood, among others are structural properties that need to be defined. Moreover, other operations performed during evolution, such as the individuals selection at local level, individuals replacement, synchronous or asynchronous individuals updating, migration policies, among others, are also implicitly modified by cGAs structural characteristics.

In Figure 2.2 a common cGA toroidal grid is shown. Several authors have reported that different lattice shapes induce different levels of selection pressure [31, 32, 33]. In [31], an analysis on the effect of the population topology and neighbourhood configuration is carried out. In order to establish a numerical relation between both, neighbourhood and topology, the dispersion pattern of p points (each individual position) centred at (x_0, y_0) was calculated. Dispersion was used as a measure because other possible measures, such as the radius of a circle, would give the same value for different population topologies. Thus, the neighbourhood and topology radii are calculated as follows:

$$D = \sqrt{\frac{\sum (x_i - \bar{x})^2 + \sum (y_i - \bar{y})^2}{p}} \tag{2.1}$$

where $(\bar{x}) = \frac{\sum_{i=1}^{p} x_i}{p}$ and $(\bar{y}) = \frac{\sum_{i=1}^{p} y_i}{p}$. Hence, the ratio between the neighbourhood

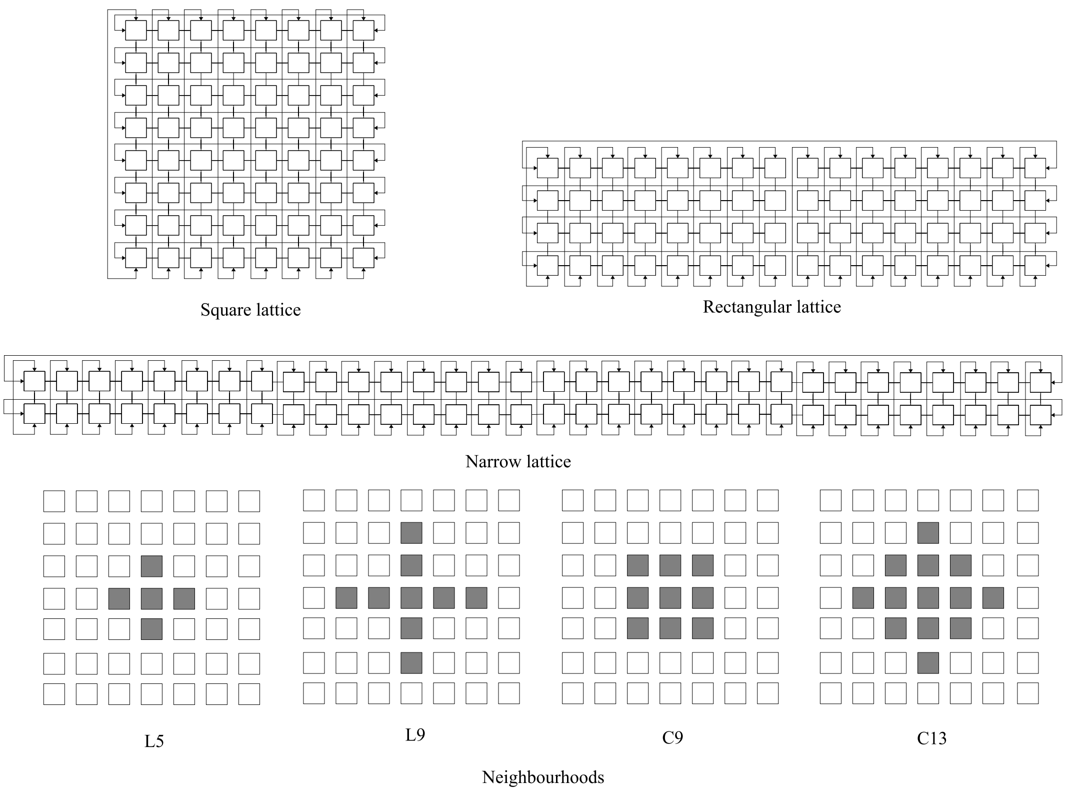


Figure 2.3: Examples of Lattices and Neighbourhoods Shapes

and the grid is given by:

$$NGR = \frac{D_{neighbourhood}}{D_{grid}} \tag{2.2}$$

this measure is known as the neighbourhood to grid ratio (NGR). Different neighbourhood shapes and sizes as well as different configurations of population topologies, as those shown in Figure 2.3, provide a distinct NGR.

A square lattice combined with a L5 neighbourhood is a configuration commonly used in cGAs implementation. In Figure 2.4, the NGR is drawn for different population sizes with a L5 neighbourhood. Three lattice shapes are considered: square, rectangular and narrow. The NGR curves show higher ratios for square grids while lower ratios are obtained for narrow topologies. This ratios difference is tightly related to the selection pressure that cGAs structural characteristics induce during evolution. In order to appreciate this effect, another concept needs to be explained: the take-over time or the growth rate of the best individual. The take-over time reflects how long it takes for the best individual to spread its solution throughout the whole lattice, applying only local selection. Thus, longer take-over times represent lower selection pressure and therefore more explorative behaviour. On the contrary, shorter take-over times correspond to

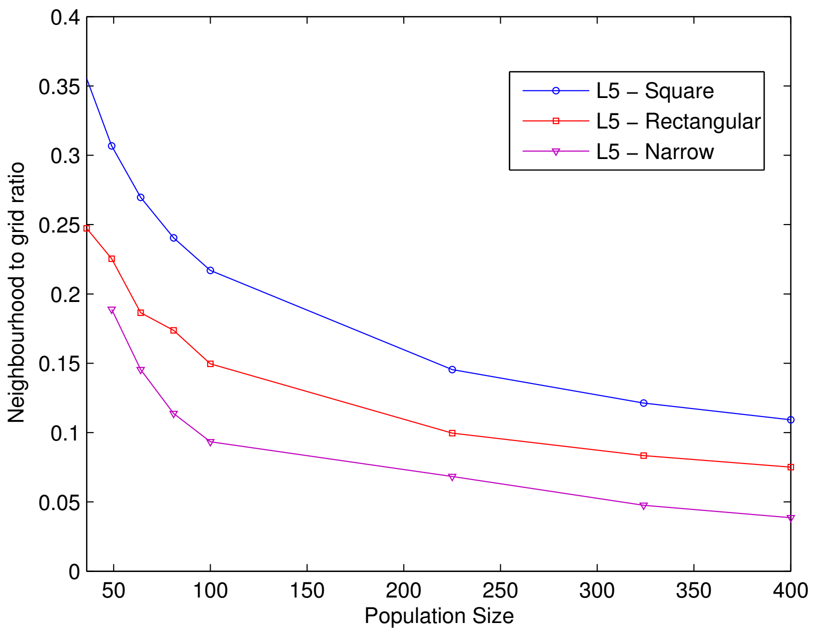


Figure 2.4: Neighbourhood to grid ratio for different population sizes and shapes with a L5 neighbourhood

higher selection pressure, equivalent to a more exploitative search.

A meta-heuristic that makes use of a set of different neighbourhood configurations (shape and size) during the search process is known as variable neighbourhood search (VNS). A solution space is initially generated either randomly or based on specific problem constraints; solutions are locally evaluated and new ones generated within limits of the local neighbourhood. Exploration takes place among distant neighbourhoods considering the best current solution and movements or jumps among those neighbourhoods are performed only if that solution improves. VNS has similarities with cGAs in terms of the different levels of selective pressures that can be induced through the configuration of the local neighbourhood. Dynamically modifying the local neighbourhood configuration during the search in cGAs would be a form of VNS. VNS is easily extendible to tackle large size problems, for more applications of this technique the reader is referred to [34, 35, 36, 37].

2.4.1.1 Theoretical take-over times

A Belgian mathematician Pierre Verhulst in the 19th century, studied the logistic growth model to describe biological systems under conditions of limited resources. Having limited resources holds for the growth rate of the best individual in a panmictic population. Considering a population size *popsize* and a number of copies of the best individual at a discrete time t given by $N(t)$. The growth rate is expressed through the following discrete recurrence model:

$$\begin{gathered} N(0) = 1, \\ N(t) = N(t-1) + p_s popsize N(t-1)\left(1 - \left(\frac{1}{popsize}\right) N(t-1)\right) \end{gathered} \tag{2.3}$$

where p_s is the selection probability for an individual and $N(t)$ can be approximated by the continuous logistic equation [38]:

$$N(t) = \frac{popsize}{1 + \left(\frac{popsize}{N(0)} - 1\right) e^{-\alpha t}} \tag{2.4}$$

where α is the p_s probability. Although similar curves are obtained by structured populations, their behaviour is not exponential but rather polynomial as suggested by Spiessen and Manderick in [39]. Thus, structural characteristics need to be included in the growth rate mathematical model for fine or cellular GAs. Considering an upper bounding case where the best individual in the neighbourhood is always selected to replace the current individual, that is $p_s = 1$. Having a population size *popsize*, with $\sqrt{popsize} \times \sqrt{popsize}$ square topology and a local neighbourhood with radius r. The best individual growth rate is described by:

$$\begin{gathered} N(0) = 1, \\ N(t) = N(t-1) + 4r^2 t - 2r(r+1), \quad 0 \le t \le \frac{\left(\sqrt{popsize} - 1\right)}{2}, \\ N(t) = N(t-1) - 4r^2 t + 4r\sqrt{popsize} - 2r(r+1), \quad t \ge \frac{\left(\sqrt{popsize} - 1\right)}{2} \end{gathered} \tag{2.5}$$

that in its closed form remains as:

$$\begin{gathered} N(t) = 2r^2 t^2 + 2r(2r+1)t + 1, \quad 0 \le t \le \frac{\left(\sqrt{popsize} - 1\right)}{2}, \\ N(t) = -2r^2 t^2 + 2r\left(2\sqrt{popsize} - 3r - 1\right)t + 1, \quad t \ge \frac{\left(\sqrt{popsize} - 1\right)}{2} \end{gathered} \tag{2.6}$$

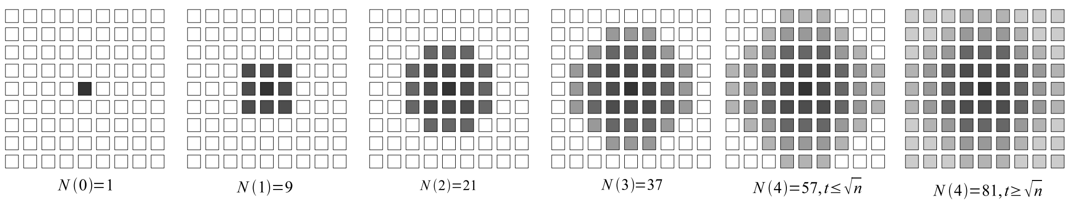


Figure 2.5: Theoretical growth rate in a square toroidal grid with a Moore local neighbourhood and selection probability $p_s = 1$

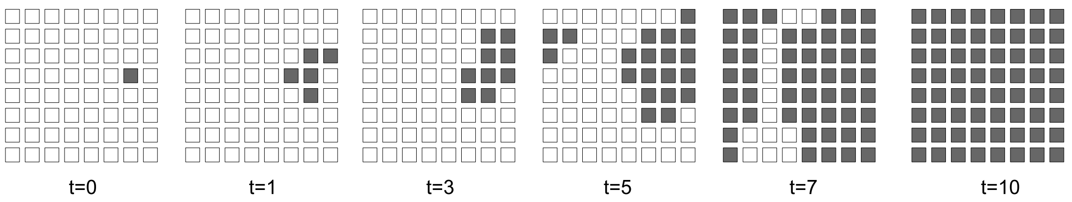


Figure 2.6: Probabilistic growth rate in a square toroidal grid with a Von Neumann local neighbourhood and selection probability p_s

In Figure 2.5, the growth rate for a population of 81 individuals with a local Moore neighbourhood is presented.

If p_s is now variable, and each individual has a different probability for selection: p_0 central individual probability and $p_1, p_2, p_3, p_4 ... p_8$ each other individual probability in a Moore neighbourhood. Modelling the exact recurrence becomes very difficult. In order to obtain simple models, the growth rate is described as the expansion of a rotated square, where the side length is given by $s = \sqrt{N(t)}$, and the half diagonal of the rotated square by $d = \sqrt{\frac{N(t)}{2}}$. Thus, individuals growing with variable probability p_i is contained inside the rotated square and is given by the following recurrence:

$$\begin{aligned} N(0) &= 1, \\ N(t) &= N(t-1) + 4p_i\sqrt{\frac{N(t-1)}{2}}, \quad N(t) \leq \frac{popsize}{2}, \\ N(t) &= N(t-1) + 4p_i\sqrt{popsize - N(t-1)}, \quad N(t) > \frac{popsize}{2} \end{aligned} \tag{2.7}$$

Finding a closed form of this model is very difficult as indicated by Tomassini [32]. In Figure 2.6 a probabilistic growth rate is shown as an example of the difficulty it represents for mathematical modelling.

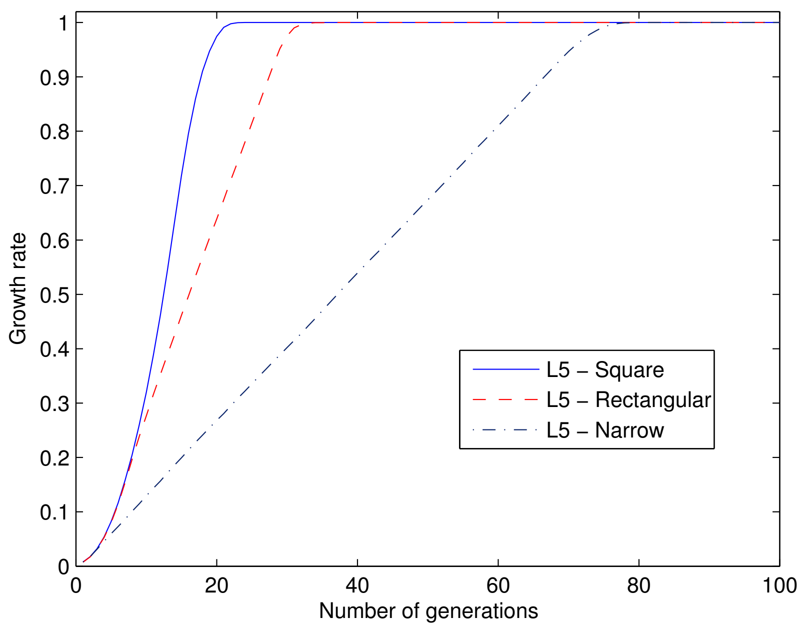


Figure 2.7: Average growth rates for the best individual on square, rectangular and narrow lattices with L5 neighbourhood. The average growth rate is calculated based on sets of 50 experiments and applying only local selection.

2.4.1.2 Empirical take-over times

An experimental set-up to empirically determine the take-over time for different lattice shapes is carried out. Binary tournament local selection is implemented and 50 experimental samples per configuration case were performed. In Figure 2.7 corresponding growth rate curves are presented considering a population size of 400 individuals, with the following grid shapes and sizes: 1) $\sqrt{400} \times \sqrt{400}$ square, 2) $10 \times \frac{400}{10}$ rectangular and 3) $4 \times \frac{400}{4}$ narrow. The difference among take-over time curves is evident. Narrow lattices provide slower diffusion of the best individual solution throughout the array and lower selection pressure, therefore a more explorative search. Instead, square lattices quickly spread the best individual solution, in a less number of generations, inducing high selection pressure, thus a more aggressive or exploitative search. In [6] an approach to dynamically change the population topology during evolution is proposed with positive results. This work is discussed in more detail in the next chapters.

Cellular GAs structural properties are subjected to further investigation in this thesis. One main difference between cGAs and panmictic GAs or distributed GAs is their structured population, and the consequent added parameters that can influence the evolutionary process based only on cGAs structural characteristics. In the next subsection, the speedups topic is discussed.

2.4.2 Speedups

A controversial topic has been the speedups that dGAs and cGAs can achieve. Yet, speedup is an accepted measure for efficiency in parallel algorithms. Calculating the ratio between the average execution time of the best sequential GA or other non evolutionary technique and the average execution time of a parallel GA running on a number of processors provides its speedup. Sub-linear speedups indicate the ratio is smaller than the number of processors for the PGA execution. Linear speedups indicates this relation is equal while super linear speedups indicate this ratio is larger than the number of processor elements [13].

A strong speedup measurement would involve a comparison against the best sequential algorithm, which would be indeed very difficult to find [40]. Considering a serial GA performance as a reference together with a stop condition that evaluates the same solution quality for all experimental samples, would allow to more fairly compare, in speedups terms, serial and parallel approaches.

Three sources for PGAs speed ups have been identified: 1) dividing the search space in several areas for exploration carried out by different processors, 2) reducing the processing load among processors through dividing the population among them and 3) genetic operators dealing with reduced data structures [41].

In [41] an analysis of dGAs speedups, with panmictic sub-populations, was presented. In part of it, a comparison versus a single panmictic GA performance resulted in excessive speedups. Yet, when the distribution of sub-populations was carried out on several processors, a comparison versus a single processor implementation showed that speedups were still obtained but these were moderate and perhaps more realistic.

This analysis is extended to consider different migration frequencies and synchronous or asynchronous migration among sub-populations. In [40], Alba et al. compared the speedups obtained for a set of test problems. Eight times super linear speedups were obtained in both approaches, when using a maximum of eight sub-populations with different migration frequencies. Lower migration frequencies together with asynchronous communications among sub-populations provided the highest speedups. These results were based on implementing the same parameters on each sub-population. In the next subsection, synchronism in PGAs is discussed.

2.4.3 Synchronism

Synchronism in parallel GAs is assessed in two arenas. In distributed PGAs with structured sub-populations, synchronism of individuals migrating among sub-populations has been investigated and compared against distributed parallel GAs with panmictic sub-populations. On the other hand, in cellular GAs, individuals updating can be performed in synchronous or asynchronous ways.

2.4.3.1 In Distributed GAs through Migration

Alba et al. investigated the effect of synchronism in the migration of individuals between sub-populations. Initially, two scenarios were considered: 1) having only panmictic sub-populations, 2) having only cellular sub-populations [42, 43]. A third scenario had later on been investigated: having a mixture of panmictic and cellular sub-populations. These approaches were evaluated not only from an algorithmic perspective but also a parallel platform was implemented [21, 12].

In [42], a MIMD (Multiple Instruction Multiple Data) implementation for homogeneous sub-populations is used. Several migration frequencies were defined as multipliers of the population size. In an homogeneous approach, a frequency of zero represents several panmictic or cellular populations evolving independently. Three problems were tackled: the generalized sphere problem, the subset sum problem, and the training of a neural network. For a fair comparison, the same configuration for genetic operations had been applied. In general, results showed a superior performance of asynchronous approaches with lower migration frequencies. In this regard, cellular sub-populations achieved significantly better efficacy than panmictic distributed PGAs. Speedups were also analysed. Lower migration frequencies reported linear speedups for cellular sub-populations and super-linear for panmictic ones.

To continue this research, Alba et al. extended the set of problems to target a wider variety of characteristics that can be found on real world problems, such as multi-modality, deceptiveness and epistasis [43, 44]. For panmictic sub-populations, a steady state reproductive cycle was implemented. This means only one offspring is reintroduced every generation if it provides a better solution to the problem. Apart from the influence of migration frequency, random selection versus best migrants selection were also compared. Similar to previous results, the best performance in terms of hit rate and number of evaluations was achieved through low frequencies and random selection of individuals for migration. Cellular dGAs outperformed panmictic dGAs in dealing with deceptiveness but performed similarly when tackling multi-modality and epistasis. Moreover, cellular sub-populations responded better to higher migration frequencies. Speedups were also assessed, the migration frequency was directly related to higher speedups in both panmictic and cellular approaches. Super-linear speedups were achieved by panmictic dGAs.

The third case considering a mixture of panmictic and cellular sub-populations was studied in [21]. Using a multi-platform approach, heterogeneous distributed PGAs were compared to results obtained by homogeneous ones. The main conclusion in this study was that super-linear speedups can be achieved also through heterogeneous computing, similar to the homogeneous approaches described above. Time measures were assessed in discrete (number of generations) and CPU real-time.

2.4.3.2 In cellular GAs through Updating Policies

Synchronism in cGAs refers to the way individuals are updated. In [45] the following updating policies were analysed: 1) line by line or line sweep, 2) fixed random sweep, 3) new random sweep and 4) uniform choice. Each of these updating policies can be implemented synchronously or asynchronously. Synchronous updating means a new entire population is generated from current individuals. Contrary to asynchronous updating that replaces an individual every time it evolves; thus individuals in one generation have evolved from offspring of the same generation.

Line by line or line sweep is the easiest way individuals can be updated in a toroidal grid topology. Following individuals sequential locations, either by row or by column, updating is performed. Fixed random sweep consist in randomly choosing an individual without replacement (each cell can be chosen once every generation) with uniform probability. A fix updating distribution of individuals is used every generation, contrary to new random sweep that in each generation applies a different random distribution. The last policy, uniform choice, randomly chooses, with uniform probability, an individual with replacement. That means a cell can be updated more than once in the same reproductive cycle.

In [32], synchronous and asynchronous updating policies in cGAs were further investigated. Initially, mathematical models of the take-over times were provided for linear and torus like cellular structures. Synchronous and asynchronous cases were evaluated in both topologies considering each updating policy case.

Synchronous updating policies presented slower growth rates, therefore a more explorative search; followed by the asynchronous uniform choice updating. Asynchronous new random sweep and line sweep showed faster growth rates but not as the fastest of a panmictic population [46]. The main conclusion was that panmictic GAs are more exploitative than cellular GAs independently of the updating policy.

Synchronous and asynchronous policies were evaluated on a variety of combinatorial and continuous problems. In combinatorial problems, asynchronous updating outperforms synchronous updating in terms of the average number of generations. Instead, synchronous updating achieved higher hit rates. All algorithmic approaches were configured without considering problem specific parameters or operations at local level. In continuous problems, overall performance results were not as clear as with combinatorial problems. Asynchronous updating performs better not only in terms of the solution accuracy, but also improved hit rates and convergence times were achieved in some cases. Yet in general, synchronous updating obtained the best hit rates.

For a complete analysis on growth rates models for asynchronous updating policies the reader is referred to [32, 46]. In the next subsection an empirical comparison between panmictic and cellular GAs is presented.

2.4.4 Panmictic vs Cellular GAs

In this section an empirical study to situate cellular GAs with respect to standard or panmictic GAs is presented. In Appendix A, benchmark problems evaluated in this section are detailed. Two competitive versions of standard GAs are implemented: the steady state GA (ssGA) and the generational GA (genGA). The name steady state comes from the idea of introducing small changes, in terms of new genetic material, every generation. Hence in $(\mu + 1)$ GA, a single offspring is introduced into the population every generation. In Algorithm 3 a ssGA pseudocode is presented. The main reproductive cycle consists of selecting two parents out of the entire population, and after recombination and offspring mutation, only one child replaces an individual in the current population. The number of offspring that replace current individuals is not limited to one but to a small number, in order to smoothly modified the population diversity.

Algorithm 3 Steady State GA

```
1: procedure SSGA
2:     $(x) \leftarrow random\,(x_0)$                                ▷ initial population
3:     $(f) \leftarrow evaluation\,(x)$                              ▷ evaluation
4:     while $k \leftarrow 1, generations,$ or $\bar{f} <= threshold$ do
5:         $(f_1, f_2, x_1, x_2) \leftarrow selection\,(f, x)$      ▷ selection from a panmictic population
6:         $(x'_1, x'_2) \leftarrow recombination\,(x_1, x_2)$      ▷ parents recombination
7:         $(x''_1, x''_2) \leftarrow mutation\,(x'_1, x'_2)$       ▷ offspring mutation
8:         $(f_{new}) \leftarrow evaluation\,(x''_1, x''_2)$        ▷ offspring evaluation
9:         $(f, x) \leftarrow replacement\,(f_{new}, x_{new})$      ▷ individuals replacement
10:    end while
11: end procedure
```

The second standard GA that has been implemented is the generational GA. The main difference between ssGAs and genGAs is the use of a temporary array to store new offspring until a population of size λ is created and replaces just part or the entire current population. The maximum size of the temporary population is the size of the population. In Algorithm 4 a pseudocode for the generational GA is presented. The main reproductive cycle consists in selecting the parents from a panmictic population and once the offspring are evaluated, one of them is stored in a temporary array, the cycle is repeated until a temporary population of size $\mu = \lambda$ is created.

In the experimental set up, the following constraints are evaluated:

- A population size of 400 individuals is used in most problems. Due to their size, GPS and MTTP problems are tackled using a population size of 64 and 100 individuals respectively.
- One hundred independent runs are carried out per experimental case.
- A problem specific threshold based on the population average fitness is used as a stop condition.
- A limit of 500 generations is used in most problems, with the exception of the Langerman function, the SLE and the MMDP problems with a limit of 700 generations.

Table 2.1: Convergence time[1], hit rate[2] and results accuracy[3] for continuous and the FMS problems. Comparing performances of steady state, generational and cellular GAs

GA	Rastrigin	Griewank	Langerman	FMS
Cellular - BT	136.69 ± 12.42	371.56 ± 53.59	291.04 ± 103.90	210.78 ± 58.76
	100%	55%	21%	41%
	$1.36 \times 10^{-4} \pm 5.07 \times 10^{-5}$	$5.77 \times 10^{-7} \pm 2.86 \times 10^{-7}$	$-1.49 \pm 5.2 \times 10^{-4}$	0.54 ± 0.56
Generational - BT	192.27 ± 86.80	–	150.00 ± 137.17	–
	98%	0%	2%	0%
	$4.16 \times 10^{-4} \pm 6.57 \times 10^{-5}$	$1.06 \times 10^{-3} \pm 2.54 \times 10^{-4}$	$-1.49 \pm 4.02 \times 10^{-5}$	113.06 ± 332.77
Generational - RW	–	–	–	–
	0%	0%	0%	0%
	0.17 ± 0.02	0.05 ± 0.01	$-2.3 \times 10^{-3} \pm 4.5 \times 10^{-3}$	123.86 ± 279.36
Steady State - BT	360.70 ± 76.34	423.27 ± 43.54	–	–
	71%	65%	0%	0%
	$4.05 \times 10^{-4} \pm 8.15 \times 10^{-5}$	$8.85 \times 10^{-5} \pm 1.25 \times 10^{-5}$	-0.77 ± 0.26	255.00 ± 612.82
Steady State - RW	–	–	–	–
	0%	0%	0%	0%
	0.62 ± 0.03	0.54 ± 0.04	$-6.74 \times 10^{-5} \pm 4.8 \times 10^{-4}$	646.53 ± 749.34

1 Convergence time is measured as the average number of generations for successful experiments. Corresponding standard deviation is included after $\pm$ symbol.

2 Hit rate is presented as the percentage of successful experiment out of the total number of experiments which is one hundred samples.

3 Results accuracy is shown as the average fitness score threshold achieved by successful experiments or the final average fitness score threshold for those scenarios where hit rate is 0%. The standard deviation is included after $\pm$ symbol.

Table 2.2: Convergence time[1], hit rate[2] and results accuracy[3] for real and combinatorial problems. Comparing performances of steady state, generational and cellular GAs

GA	SLE	GPS	MMDP	MTTP
	297.28 ± 72.56	73.12 ± 41.83	431.31 ± 54.44	259.65 ± 44.16
Cellular - BT	39%	57%	89%	99%
	2.60 ± 0.71	$0.99 \pm 4.08 \times 10^{-8}$	-25.00 ± 0.00	193.00 ± 0.00
	666.86 ± 126.76	398.29 ± 180.34	–	–
Generational - BT	12%	27%	0%	0%
	3.12 ± 1.21	$0.99 \pm 1.96 \times 10^{-4}$	-20.61 ± 0.43	582.69 ± 164.14
	556.02 ± 204.69	–	–	–
Generational - RW	37%	0%	0%	0%
	3.78 ± 0.19	$0.98 \pm 3.0 \times 10^{-3}$	-13.72 ± 0.45	6291.55 ± 962.58
	255.00 ± 63.00	110.00 ± 85.19	–	361.92 ± 62.29
Steady State - BT	3%	26%	0%	56%
	3.80 ± 0.10	$0.99 \pm 3.03 \times 10^{-7}$	-9.64 ± 0.31	193.92 ± 2.10
	–	163.63 ± 95.97	–	–
Steady State - RW	0%	33%	0%	0%
	2504.98 ± 432.99	$0.99 \pm 3.02 \times 10^{-7}$	-7.33 ± 0.74	31850.38 ± 1823.38

1 The convergence time is measured as the average number of generations for successful experiments. The corresponding standard deviation is included after ± symbol.

2 The hit rate is presented as the percentage of successful experiment out of the total number of experiments which is one hundred samples.

3 The results accuracy is shown as the average fitness score threshold achieved by successful experiments or the final average fitness score threshold for those scenarios where hit rate is 0%. The standard deviation is included after ± symbol.

Algorithm 4 Generational GA

```
1: procedure SSGA
2:     $(x) \leftarrow random\,(x_0)$                                    ▷ initial population
3:     $(f) \leftarrow evaluation\,(x)$                                  ▷ evaluation
4:     while $k \leftarrow 1, generations,$ or $\bar{f} <= threshold$ do
5:         for $i \leftarrow 1, \lambda$ do                              ▷ $\lambda \leq populationsize$
6:             $(f_1, f_2, x_1, x_2) \leftarrow selection\,(f, x)$   ▷ selection from a panmictic population
7:             $(x'_1, x'_2) \leftarrow recombination\,(x'_1, x'_2)$     ▷ parents recombination
8:             $(x''_1, x''_2) \leftarrow mutation\,(x'_1, x'_2)$        ▷ offspring mutation
9:             $(f_{new}) \leftarrow evaluation\,(x''_1, x''_2)$         ▷ offspring evaluation
10:            $(f_{temp}, x_{temp}) \leftarrow replacement\,(f_{new}, x_{new})$   ▷ temporary allocation
11:        end for
12:        $(f, x) \leftarrow (f_{temp}, x_{temp})$                     ▷ individuals replacement
13:    end while
14: end procedure
```

The cGA is implemented on a square toroidal topology with a local Von Neumann neighbourhood consisting of four individuals plus the central one. Local selection is performed with the central individual always as the first parent while the second is selected through binary tournament from individuals in the neighbourhood. On the other hand, panmictic GAs are implemented using two selection methods: binary tournament (BT) and roulette-wheel (RW). BT randomly selects two individuals from the entire population (or from the neighbourhood in cGAs case) as parents for mating. In contrast, RW selection, as it name implies, defines a slot for each individual based on its fitness score. It means, better individuals have larger slots in comparison to individuals with low fitness scores. Due to the tight limit in the number of generations, the steady state GA is executed as $\left(\mu + \frac{\mu}{\sqrt{\mu}}\right)$. Thus, 50 individuals, for a population of 400 individuals, are replaced every generation. The generational GA is experimentally assessed with $\mu = \lambda$.

Tables 2.1 and 2.2 show performance results obtained in terms of the average number of generations, hit rate and results accuracy. The stop condition in all cases is the maximum number of generations or when the average fitness score reaches a problem's specific threshold, see Appendix A for more details on each problem.

The superior performance of cellular GAs is noticeable. When tackling continuous problems, the ssGA and genGA through RW selection perform very poorly, in comparison to the cGA and both panmictic GAs through BT selection. Cellular GAs achieve better performance for the Rastrigin and the Langerman functions. However, a better hit rate is achieved by the ssGA through BT when tackling the Griewank function. For the FMS problem presenting strong epistasis, only the cellular GA is able to solve it, although with a low hit rate of 41%. Comparable results are obtained for the SLE problem; with similar performance achieved by the genGA through RW selection in terms of hit rate, 39% (cGA) and 37% (genGA). However, the later almost double the average number of generations. For the rest of the problems, the GPS, the MMDP and the MTTP, cGAs outperformed panmictic approaches with the experimental constraints here implemented; a clear example is the MMDP where none of the panmictic GA versions converge to the global optimum.

In the next section, a review of several architectures based on panmictic and cellular GAs are revised.

2.5 Hardware Implementation of GAs

From the theoretical to the real world, GAs feasibility for hardware implementation has also been investigated. From the Very Large Scale Integration (VLSI) point of view, GAs possess important characteristics that makes them suitable for hardware implementation targeting real time applications.

Not only problem specific hardware architectures based on GAs have been developed but also evolutionary design of electronics has been targeted through GAs. This relatively new research arena is known as Evolvable Hardware (EHW) [9]. Parallelized versions of GAs have been used as optimizers and adapted for hardware implementation targeting real-time performance. Among problems solved by parallel GAs are: the image registration problem [47, 48, 49], the disc scheduling problem [50] and the GPS attitude parameters determination problem [5, 51, 52, 53]. On the other hand, in EHW, Sekanina highlights the difference between solving a problem using a hardware implementation of an EA, and to evolve a digital or analog circuit in order to obtain the best configuration bit stream [9]. Some of the work related with EHW is reported in [54, 55, 56, 57, 58], these articles are concerned to evolutionary techniques targeting FIR filters design and linear transformations.

In [50], a cGA architecture was developed targeting the disc scheduling problem. It consists in finding the best attending tasks order to reduce the access time per request. In this architecture, a cellular GA encodes up to 32 queued requests ordered by a fitness function that minimizes latency and search time per request. The architecture achieved an scheduling time of up to 2 milliseconds per access request and 4 milliseconds of searching time per request. Thus, timing constraints were fulfilled. Ordered based crossover and mutation operations were performed. Ordered based crossover randomly selects chromosome positions in one parent, finds these positions in the second parent and copies corresponding alleles to the offspring. Thereafter, a reordering policy is applied, for selected genes to maintain their positions in the offspring. Ordered based mutation selects certain chromosome positions and mutate the alleles.

An image processing architecture was developed to tackle the image registration problem that presents real-time constraints [47, 48]. First, in [47] an architecture for image registration using a cellular GA was proposed. The image registration problem consists in matching a 2-D captured image to a reference image. A transformation between both images is necessary. The fitness function measures the number of corresponding pixels between captured and reference images. Once a perfect match has been found, the transformation encodes the object's position and orientation. Each transformation parameters were encoded in 6 bits. At algorithmic level, an extra step was added to the canonical cGA procedure defined by Tomassini [32]. Each transformation parameter is incremented or decremented in one unit for promotion through hill climbing, while the best individual is always kept. Results showed that real-time constraints are fulfilled up to a tenth of a second to match the images, corresponding approximately to 35 generations for 64×64 images.

In [48], an improved cGA architecture for image processing combining data compression and image registration is presented. For the image registration, a cGA determines an affine transformation for the image coordinates with respect to a reference image. The fitness function minimizes the error between the transformed and the referenced images. Then, the Discrete Cosine Transform (DCT) using a data compression algorithm is applied in order to reduce the memory storage requirements. In terms of speed, processing one chromosome took up to 2 milliseconds with limited accuracy for 64×64 images.

A cellular GA architecture tackling the GPS attitude determination problem was developed by Xu et al. In this thesis this problem, at algorithmic level, is further investigated from a fault tolerant perspective. An extended analysis and results of the proposed approaches are included in Chapter 3. In [5, 51] a method to determine the attitude parameters of a vehicle based on the Global Positioning System (GPS) technology and the Ambiguity Function Method (AFM), is presented. The attitude parameters are determined by the vector difference between GPS receivers attached to the vehicle.

Binary chromosome representation of 32 bits is used. The azimuth angle is encoded in the first 14 bits, the elevation angle in the next 10 bits and the base line length in the last 8 bits. The minimum steps for each parameter are 0.022, 0.030 degrees and 0.78 millimetres respectively.

In [51] a comparison among a panmictic GA, a panmictic GA with fine and coarse search stages, and a cellular GA with fine and coarse stages was carried out. Different array sizes were tested in order to find the best population size to solve the problem. Algorithmic performances were measured in terms of the average number of generations, and the hit rate together with the results accuracy. Results showed that a cGA with population size of 25 outperforms both panmictic GAs approaches.

Stefatos et al. extended the GPS attitude architecture presenting a novel approach from a fault tolerance perspective to deal with Single Hard Errors (SHEs) and implicitly with Single Event Upsets (SEUs), both are Single Event Effects (SEEs) subclasses [52]. Erroneous bit flipping in data registers, temporary (SEUs) or permanently (SHEs) is the effect of this kind of faults. If faulty registers correspond to critical data, the system's functionality fails. In the proposed architecture, a faulty scenario considering stuck at zero faults at fitness score registers was targeted. In the cGA, individuals with faulty fitness scores are not selected and therefore their solution is not spread throughout the population.

The fault tolerant architecture is a double layer approach, each layer implements a cellular GA. A computational layer to determine the GPS attitude parameters and a control layer to determine the best configuration of individuals or Processor Elements (PEs) to overcome a faulty scenario. Several experiments were carried out, first only the computational layer was tested in order to observe how the cGA deals with the faults. Then the control layer was executed and a performance comparison is carried out. Results showed a significant improvement in the system's performance when the control layer was active, in the worst case scenarios with 30% and 40% faulty PEs. Although, using the control layer allows an improvement in the system performance, there is a significant increment in the use of hardware resources. Another disadvantage is that the control layer was considered free of faults.

Finally, in [53] a high performance hardware architecture for the GPS attitude determination problem was presented; emphasis was paid to the speed, power consumption and hardware usage. Thus, a Coordinate Rotation Digital Computer (CORDIC) module was implemented to calculate the trigonometric functions required by the fitness function [59]. Results fulfil real-time constraints due to the simplified arithmetic units. This subsection provided a first glance to the fault tolerant arena. In the next section a more detailed review on this topic is carried out.

2.6 Fault Tolerance

The operation of electronics systems needs to be reliable. However, scenarios exist where systems are subjected to tough environmental conditions or where human life could be at risk. For example, systems operating in critical environments such as land, water, or air craft systems, outdoor operating systems, or medical equipment which operation not only involve risks but also human life is compromised, either directly or as a consequence of the results provided by an equipment.

In the first part of this research, the GPS attitude determination problem (see Chapter 3), is the case study tackled from a fault tolerant perspective. The GPS attitude module is central to navigation systems. In air navigation, electronic circuits of an aircraft are prone to radiation due to the effect of high charged particles that could affect them at an operational or physical level. A complete analysis of this kind of faults known as Single Event Effects (SEEs) is presented in [60].

SEEs have been researched for more than three decades, in which several approaches at different system levels have been applied. Among them, the development of SEE hardened device technologies has relieved some of the problems. However, several disadvantages still remain, such as high power consumption, increasing usage of hardware resources, loss of performance, reduced availability and high costs. Thus, developing new techniques to deal with SEEs is still necessary not only because of the very elaborated processing on today's systems but also the sensitivity as regards to smaller fabric sizes.

There are several types of SEEs, in particular Single Hardware Errors (SHEs) and implicitly Single Event Upsets (SEUs) are investigated in this research. SHEs induce a lasting change on the operation of a system by permanently stuck at logic zero or one critical data registers or memory allocations. SEUs are transients or temporal state changes caused by the energy induced in a device by an ionizing particle such as a cosmic ray or a proton. The functionality of the system is retrieved after data rewriting or a system reset.

Developing SEE tolerant systems is nowadays carried out from a functional rather than a physical perspective, mainly because the energy induced by SEE causes functional impacts by propagating in the worst case scenario to all modules in a system. This condition has guided SEE analysis to an algorithmic level by identifying the main functions and critical data that need to meet operational constraints when SEEs occur.

Several approaches have been developed in order to deal with SEEs. Among those, Triple Modular Redundancy (TMR) with voting, triplicates one or more system modules [61, 62]. However, the voting module of a TMR system is considered free of faults, but SEE can still affect it. Moreover, increasing system's reliability through this method requires an exhaustive use of hardware resources.

Another hardening technique, known as DICE (Dual Interlocked Storage Cell), is applied at circuit level [63]. It duplicates the storage latch and uses state restoring feedback which is more compact and add less delays than TMR systems. Yet, an important disadvantage of this technique is the difficulty in its application on highly dense devices due to the significant area overhead.

A widely used combination in VLSI (Very Large Scale Integration) technology is the FPGA (Field Programmable Gate Array) using SRAM (Static Random Access Memory). This reconfigurable platform has dramatically increased its density in the last two decades, making it also very attractive because of small fabric sizes. However, SRAM is highly sensitive to induced radiation and therefore prone to SEE, in particular SEU, SHE and their multiple counterpart: MBU (Multiple Bit Upset) [64].

The effect of sizing reduction, low power requirements and high frequencies negatively affect devices reliability. Combining these trends of new technologies has increased the probability of SEEs. Operating on small interconnections at higher frequencies increases the number of errors due to timing constraints violations. On the other hand, smaller transistors operating at low power voltages increase their sensibility to neutron and alpha particles, and therefore higher error rates are observed [65]. Because of the advances in manufacturing processes, permanent faults occurring as a consequence of physical damage, have been reduced significantly. In contrast, temporary or transitory and intermittent faults have increased. Hareland et al. empirically demonstrate the negative effect of fabric size reduction for alpha particles and high energy neutrons, affecting electronics due to terrestrial radiation. For 0.25 μm technology at $2V$, the SER (Soft Error Rate) resulted in nine times increase considering SRAM or DRAM (Dynamic RAM) size from $2Mb$ to $10Mb$. In contrast, $0.18\mu m$ technology at $1.6V$ for alpha particles reported an increase of ≈ 90 to ≈ 380 SER for the same S/D RAM sizes [66].

In [67], the last reliability report of one of the main companies producing FPGAs, Xilinx, reports the following SEU and Soft Error Rates measurements in cells used as configuration memories and block RAM: For $250nm$ (Virtex family) technology node, $160FIT/Mb$ (Faults in Time per Megabyte) in configuration memory and $160FIT/Mb$ in block RAM, with an error of $\pm 20\%$. $130nm$ (Virtex-II Pro family) and $90nm$ (Virtex-4 family) node technologies imply: $386FIT/Mb$ and $261FIT/Mb$ in configuration memory; while in block RAM $655FIT/Mb$ and $438FIT/Mb$ are reported with errors of $\pm 7\%$ and $-17\%/+22\%$ respectively. One FIT is equivalent to one fault per 10^9 operation hours.

Referring to Multiple Bit Upsets for the same Xilinx families, in [68] is reported that MBUs are more likely to occur in the Virtex family in comparison to the Virtex-II and Virtex-4 families. Results showed that I/O Blocks are very sensitive to MBU and in Virtex II and 4 Pro families are nearly as sensitive as CLBs (Configurable Logic Blocks). The observed MBUs are events of more than 5 bits in the Virtex-4 family. On the other hand, in the same JPL report, due to fabric size reduction, MBUs are reported to be 27-33 times more likely to occur in the Virtex II and II Pro families than in the former Virtex family. Moreover, in $90nm$ Virtex-4 and $130nm$ Virtex II and II Pro, MBUs are 3 and 69 times more likely to occur than in $220nm$ Virtex family.

Considering the empirical results discussed above, in the next chapter fault tolerant mitigation techniques are presented and empirically assessed. The case study is the GPS attitude determination problem which is a hard real-time constrained problem. Following the remarks of the SEE criticality analysis presented in [60], as being more appropriate to deal with SEEs at software and hardware levels; several soft fault tolerant mitigation techniques are developed in this research. The proposed fault tolerant mechanisms take advantage of the algorithmic properties to tackle the problem. A cellular Genetic Algorithm is implemented and critical data structures are considered prone to SEE that can lead the system's normal operation to fail. For example, if the fitness score or the chromosomes registers are faulty, the cGA would fail in determining the attitude parameters. These techniques take advantage of the inherent structural properties of cellular GAs with a minimum increase in the computational cost.

Chapter 3

On Fault Tolerance

In this chapter, several experimental set-ups to evaluate the ability of cellular GAs to deal with faults are analysed. Specific types of Single Event Effects (SEEs) known as Single Hard Errors (SHEs) and implicitly Single Event Upsets (SEUs) are evaluated. Those kinds of faults cause permanent or temporal changes in system data that can be critical for normal operation. The operation of real-time systems based on cGAs platforms, such as the image registration, the tasks scheduling and the GPS attitude determination problems, are prone to faults due to different reasons, such as tough environmental operation conditions, highly dense fabrics, induced radiation, among others. However, fault tolerance requirements are not only necessary in systems which are susceptible to faults; but indispensable also in systems operating in safety-critical conditions, such as medical equipment, vehicles for human transportation, among others.

In [69], fundamental concepts of fault tolerant systems are introduced. Firstly, the definition of a real-time system consisting of two aspects: being logically and temporarily correct. Logical correctness refers to the proper functionality of the system while the temporal correctness refers to the fulfilment of timing deadlines. Secondly, two independent stages should be considered in a fault tolerant system: fault detection and isolation followed by fault recovery. In this chapter, the second stage is approached assuming the isolation of faulty elements. Autonomous execution of both stages is essential in fault tolerant systems [70].

A representative case study is tackled in this chapter: the GPS attitude determination problem. This problem presents a number of characteristics that makes it suitable for analysis from a fault tolerance perspective. Determining the attitude parameters of a vehicle is an essential part of its navigation system, in land, water, air or outer space vehicles. Such a system can be subjected to harsh environmental conditions, including high doses of radiation which could directly affect data critical to the system. It is a hard real-time application requiring an adequate algorithmic approach to fulfil timing constraints. It is also a safety-critical application in vehicles involving people transportation. On the other hand, a permanent failure of an attitude determination module and therefore of the vehicle's navigation system mean a loss of expensive equipment for example in unmanned vehicles.

It is also a system which operation implies people's safety or high costs in equipment not accessible for maintenance or repair, such as in unmanned vehicles.

Current research on the fault tolerance arena is a continuation of previous work in the SLI group. Xu et al. explored the suitability of cellular GAs to solve the GPS attitude determination problem fulfilling real-time requirements [5]. Thereafter, Stefatos et al. proposed a fault tolerant architecture to deal with stuck at zero faults at fitness score registers [52]. An improved high performance version of this architecture was then implemented [53]. In this research, topological characteristics, such as the neighbourhood and grid configuration; together with structurally attached operations of cGAs, such as local selection, migration and replacement policies are investigated as mitigation techniques to deal with SHEs and implicitly with SEUs.

In order to assess cGAs ability to deal with SEEs, data critical to the system are subjected to faults. In this chapter, two sections are dedicated to analyse the proposed fault tolerant techniques. In Section 3.2, SHEs/SEUs are induced at the phenotypic space in fitness score registers. Three types of faults are considered: stuck at zero, stuck at one and a combination of both, a hybrid case. In Section 3.3, SHEs/SEUs are injected in the genotypic space at chromosomes registers which encode the individuals. The same kinds of faults as in the former case are evaluated. Next, in Section 3.4 a distributed cGA approach is presented aiming to improve the results accuracy and after that to incorporate fault tolerant mitigation techniques. To begin with in the next section, the mathematical basis for the GPS attitude determination problem is explained.

3.1 Case Study: GPS attitude determination

The electronic circuits of an aircraft vehicle consist of several subsystems. Among them, the attitude control module represents the navigation core of a vehicle, attaining and maintaining its position. Hierarchically, the attitude control system is associated with several subsystems which may include different kinds of sensors, gyroscopes and support electronics. Recently, a new attitude determination technique based on Global Positioning System (GPS) technology has been developed [5]. An advantage of this technique is its simplicity, since it does not employ any elaborated or expensive systems including solar sensors or inertial elements.

In [5], a cellular GA to determine the GPS attitude parameters of a vehicle was presented. This approach is based on the Ambiguity Function Method (AFM) which is a full search technique that can deal with cycle slips but has an expensive computational cost [71, 72]. The fitness-function of this method measures the carrier phase difference between two antennas attached to a vehicle. In [51] a cGA is compared against a standard GA, in both cases, a coarse to fine search process is applied in order to overcome standstills at local optima. Simulation results were provided in order to show that cGAs presented superior performance over standard GAs, when using different population array sizes, the best performance is obtained using an array of 25 individuals.

In [52] a cellular GA to explore the feasibility of a fault-tolerant VLSI architecture targeting the GPS attitude determination problem, is presented. The performance of cGAs in the presence of SHEs and implicitly SEUs, which occur at critical data registers was investigated. Initially, faults were restricted to registers that store individuals fitness score. SHEs and SEUs are considered as changes in microelectronic devices caused by the presence of high-energy particles which may induce permanent or temporary single or multiple bit flips in registers. The fault-tolerant approach presented in [52] consists of two layers. The first implements a cGA targeting the attitude parameters determination, while the latter, which is also based on a cGA, acts as a control to handle faulty processor elements. This hardware design showed to improve the performance of the system with the inclusion of the control layer in faulty scenarios. However, implementing a second cGA layer implies a significant increase in hardware resources usage. Finally, in [53] a high-performance VLSI architecture targeting real-time operation and the reduction of hardware usage was presented. In the next section

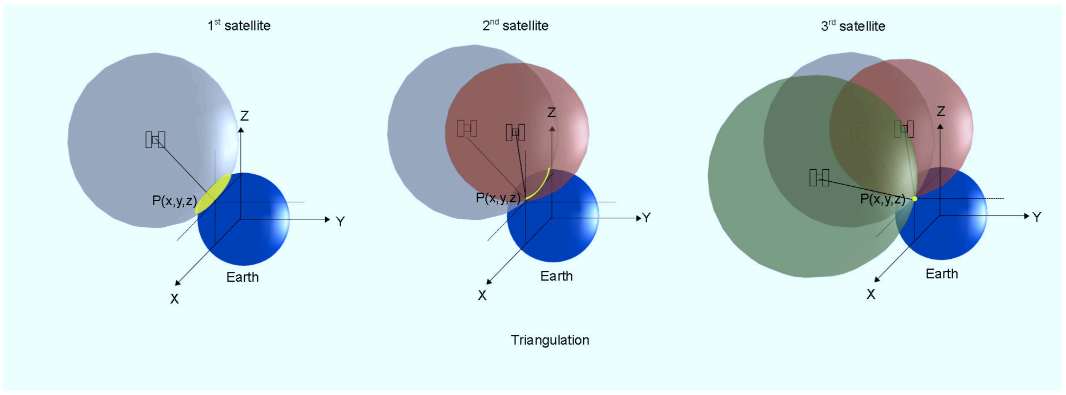


Figure 3.1: GPS triangulation

an introduction to the GPS technology is presented.

3.1.1 Brief introduction to GPS technology

GPS is a Global Positioning System used originally in the military arena, but opened later for civil use. It consists of an array of 24 satellites in six orbital planes with four satellites on each plane. Each satellite contains four atomic clocks for continuous and highly accurate repositioning and reorientation. GPS operation is based on measuring the signals travelling times, and therefore the distances, from several simultaneously observable GPS satellites to a GPS receiver. Knowing the distances between satellites and receiver, facilitates the determination of the receiver's position with accuracy within meters. The main applications for GPS technology are positioning and navigation. In [73], an introduction to GPS technology is provided and in [74] mathematical and algorithmic details are covered.

In Figure 3.1, the process to calculate the position of a GPS receiver is shown. Considering an Earth centred coordinate system, the intersection of an imaginary sphere centred at a GPS satellite position and the Earth delimits the searching area of the receiver position. A second imaginary sphere centred at a second GPS satellite position reduces the search area for positioning a GPS receiver to a plane. Finally, intersecting a third imaginary sphere centred at a third GPS satellite position provides a two points option for the receiver's position. A fourth GPS satellite measurement is necessary in order to eliminate any clock error in the GPS receiver using the time provided by a fourth satellite's atomic clock.

Calculating the distance (range) between at least four GPS satellites and a receiver is necessary to mathematically determine its position. GPS satellites use radio signals. In order to determine the total distance travelled by a signal, from satellite to receiver, the velocity of the signal and its time to arrive are required. The travel velocity is considered to be the speed of light, $300 \times 10^3 \frac{km}{sec}$. Ideally a signal travelling from a satellite located exactly overhead to a point on the Earth's surface, takes around 6×10^{-2} seconds to arrive. Yet, higher accuracy is required in GPS calculations and therefore more information is needed.

Calculating the time for signals to travel from satellites to receiver is done using a pseudo-random code which is a string of 1023 bits repeated every millisecond. Each satellite possess a unique pseudo-random code to identify itself once its signal arrives at the receiver. Only 37 codes out of the total number of possibilities (2^{1023}) are weakly correlated and can be used for this purpose [75]. The same pseudo-random code signal leaves the satellite and the receiver is delayed until both are synchronized. The pseudo-random code shifting at the receiver determines how long the signal takes to arrive.

There is another issue to calculate the position of the receiver. Timing at satellites is as accurate as possible, due to the atomic clocks each satellite is equipped with. However, receivers' clocks do not provide the same accuracy, due to obvious cost reasons. Instead, a fourth satellite's signal is acquired and the difference is calculated with respect to the other three measurements, calculating in this way the exact signals travelling time.

The GPS process described before is an ideal one. It means that no induced errors due to environmental conditions have been considered. Thus, in order to get more accurate results, error correction is implemented in GPS receivers. It was mentioned that in order to calculate the distance between satellites and receivers, the speed of light multiplies the signal's travel time. However, the speed of light occurs in vacuum conditions, and the signal travels through several layers with charged particles and water: the ionosphere and the troposphere. On each of these layers the signals slow down inducing a delay similar to having a clock error. Determining the error due to environmental factors can be carried out by modelling these conditions. However, environmental modelling is not an easy task. Instead, it is possible to compare the relative difference between two different signals: dual frequency. Yet, that method is only available in advanced receivers.

Minor timing errors can also be induced by satellites. The GPS constellation has terrain stations that frequently communicate with the satellites in order to adjust their position and orientation. This information known as *ephemeris* is updated in an almanac every receiver has when acquiring GPS signals. However, ephemeris updating does not happen every second, thus other sources for correction need to be considered. In order to improve GPS technology to deal in a much better way with those setbacks, which to date have been accepted as normal in GPS operation and also to achieve higher accuracy, the differential GPS was created.

Differential GPS has two receivers instead of one. One of the receivers is static while the other remains mobile. Both receive signals from a satellite which travel through the same environmental conditions. GPS satellites orbit more than 28×10^3 meters above the earth's surface. Thus, a distance on the earth's surface of a hundred meters, does not represent a significant change with respect to a satellite orbiting position. It is considered that the static receiver, due to its fixed and therefore well known position, will calculate an accurate position, and will provide the mobile receiver with the corrective factors. In this way, the errors mentioned before can be eliminated.

Because the static GPS receiver knows its actual position accurately. Working as a differential GPS, it calculates the times for current travelling signals and then compares their time with its own information. Then, it transfers this information to the mobile receiver to correct the calculation of its position.

The static GPS receiver does not know which satellite(s) the mobile GPS is using to makes its calculations. Therefore, the mobile receiver acquires all signals coming from all visible satellites and transmits all correction factors from those satellites to the static receiver which decides which one(s) to use. In the next subsection details of the mathematical method used to determine the attitude parameters of a vehicle are introduced.

3.1.2 Ambiguity Function Method

Recently, GPS technology has been applied in the determination of attitude parameters of a vehicle through the calculation of the correct carrier phase integer ambiguity values [71]. Several techniques have been developed in this regard, such as the Ambiguity Function Method (AFM) which is a full search method, computationally expensive but not sensitive to cycle slips that corrupt the measurement of the carrier phase due to a temporary clock loss during signals tracking [5]. The AFM determines the

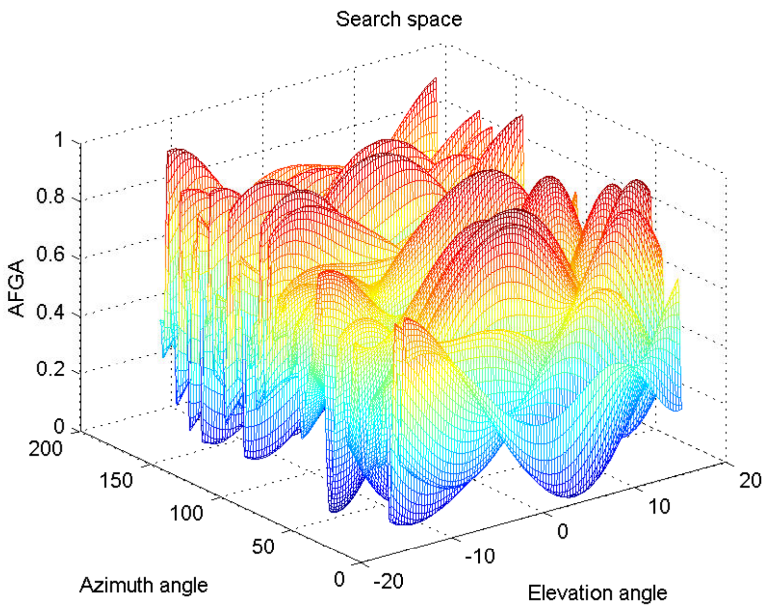


Figure 3.2: AFGA search space

correct carrier phase integer ambiguity values through the calculation of the double difference observable between satellites and GPS receivers. The AFM function which guides the search is a multi-peak, non-linear function combining trigonometric functions such as sine and cosine. GAs have shown their ability to deal with such complicated search spaces successfully. In Figure 3.2, an example of the search space is presented considering an azimuth range of 0 to 200 degrees, an elevation range of ± 20 degrees with a fixed baseline.

Figure 3.3 shows how the AFM determines the attitude parameters. The baseline vector is defined by points A and B which are the extremes of a vehicle where the GPS receivers are attached. The wave fronts arriving at those points present different true distances between satellites and receivers. The carrier phase measurement at each point is defined in terms of the true distance (ρ_A^j, ρ_B^j), the light speed (C), the satellites and receiver clock errors (dT^j, dT_A, dT_B), the ionospheric and tropospheric delays (d_{ion}^j, d_{trop}^j), the integer cycle ambiguity (N_A^j), and the GPS carrier signal wavelength, which is $\lambda = 0.19m$ for commercial GPS receivers. The equation at point A is given by:

$$\Phi_A^j = \rho_A^j + C(dT^j - dT_A) + \lambda N_A^j - d_{ion}^j - d_{trop}^j \tag{3.1}$$

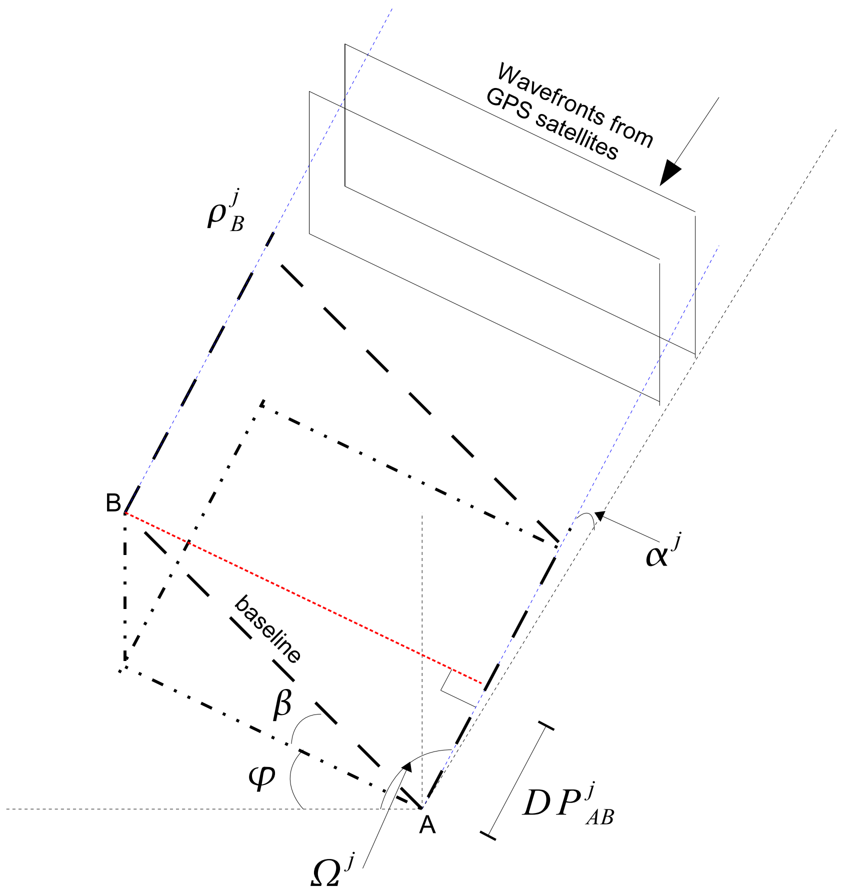


Figure 3.3: Carrier phase measure between GPS satellites and receivers

If the single difference is calculated, $D\Phi^{j}_{AB} = \Phi^{j}_{A} - \Phi^{j}_{B}$, only the satellite clock error is eliminated, but the receiver clock error remains and is coupled to the ambiguity term. The calculation of the double carrier phase difference eliminates this error. On the other hand, the ionosphere and troposphere delays are not considered because the distance between the receivers is assumed to be short, up to 50 meters. The unknown double difference value for the carrier phase is calculated as:

$$DD\Phi^{jk}_{AB}(\varphi, \beta, b) = b\left[\sin\beta\left(\sin\alpha^{j} - \sin\alpha^{k}\right) + \cos\beta\left(\cos\alpha^{j}\cos(\Omega^{j} - \varphi)\right)\right. \\ \left. - \cos\alpha^{k}\cos\left(\Omega^{k} - \varphi\right)\right] \tag{3.2}$$

Finally, the fitness function presented in [5, 51] is defined in terms of the angles between the horizontal and vertical planes and the baseline. The maximum possible value is close to 1.0 due to the induced noise:

$$AFGA\left(\varphi, \beta, b\right) = \sum_{i=1}^{m}\sum_{j=2}^{n}\cos\left(\frac{2\pi}{\lambda}\left(\frac{DD\Phi^{1j}_{AB}}{m\left(n-1\right)} - \frac{DD\Phi^{1j}_{AB}\left(\varphi, \beta, b\right)}{m\left(n-1\right)}\right)\right) \tag{3.3}$$

where $DD\Phi^{1j}_{AB}$ is the known double difference for the observable carrier phase, n is the number of satellites (4-6 satellites), and m is the number of epochs, one epoch is the time interval in which the satellites information is received by the GPS antennas.

3.2 Faults at Phenotypes

In order to assess the inherent ability of cellular GAs to deal with SHEs and SEUs that could permanently or temporarily affect the operation of a system, several parameters which directly affect the performance of their search process need to be monitored and controlled. Those parameters include the migration rate and frequency, the population size and the local neighbourhood configuration. All of them have a remarkable effect in the performance of cGAs. By appropriately controlling these parameters, the difficult search space of the GPS attitude determination problem, presenting a multiple-peaks landscape, would be effectively explored.

Fitness score registers have been identified as critical and when subjected to faults, particularly SHEs, the search process is at risk. Representing a major disadvantage to fulfil hard real-time constraints. It was mentioned before that three faulty scenarios are evaluated: stuck at zero or stuck at one bits and a hybrid case. The severity of faulty scenarios in the phenotypic space depends on the problem to optimize. For example, minimization would be more affected for stuck at zero faults because locally during selection, individuals with low fitness scores are more likely to be selected and consequently, poor solutions could spread. The opposite case occurs in a maximization case. However, in reality, a hybrid case is more likely to occur. Therefore, all possible scenarios are evaluated. In the next subsections topics like the operations specific to cGAs, the proposed algorithmic approach and the injection of faults are covered.

3.2.1 SHEs Mitigation Approach

There are several parameters that affect cGAs operation, such as the size of the grid, the size and shape of the local neighbourhood, the local selection method, among others. Initially, the study of cGAs has focused on their algorithmic performance based on empirical knowledge of those parameters. However, there has been an effort to provide mathematical models to explain their behaviour, see Subsection 2.4.1.

On the other hand, migration is a genetic operator used in distributed and cellular GAs to provide genetic diversity among sub-populations and individuals respectively [22]. Applying migration is useful to exchange genetic material among sub-populations or neighbourhoods, and to avoid a standstill during the search process. Migration policies are defined through several parameters such as the number of individuals to migrate, the migration frequency, and the migration selection and replacement rules. In [20] several migration policies are compared, results showed better performance when the best individuals are selected for migration and those replace the worst individuals in the receiving sub-populations.

In the cGA configuration proposed here, explicit migration is applied to show its ability as a mitigation technique to overcome faulty scenarios where data critical to the system are affected. Initially, highest migration rate and frequency are assessed. Highest frequency means migration occurs every generation, highest rate means all individuals migrate. Applying bounding cases serve as an indicator of cGAs performance and normally these parameters are adjusted to intermediate values. Several migration policies are evaluated and results are provided in Subsection 3.2.4.

Migration would provide every individual with the option of looking for other neighbourhoods which might have less number of faulty individuals. An individual would not benefit from migration if reproduction is performed in a neighbourhood with solutions which are worst than its current one. Therefore, several selection and replacement criteria are evaluated. In Subsection 3.2.3 details about the proposed migration criteria are presented.

An analysis on selection intensity in cGAs is presented in Subsection 2.4.1. Selection intensity or selection pressure are referred indistinctly. In cGAs, selection intensity is tightly attached to the topology in which individuals lie on, and can be controlled through the size and shape of the local neighbourhood. In [31], it is shown how the growth rate of the best individual changes according to number of individuals in the local neighbourhoods and its shape. From the fault tolerance perspective, increasing or decreasing the selection intensity through the local neighbourhood configuration would allow to sort out faulty individuals without affecting the results accuracy and the convergence time. In Figure 2.3 several local neighbourhoods named after their shape as L5, L9, C9 and C13 are drawn. L5 and L9 are cross-like neighbourhoods which include individuals at North, East, South and West. L5 only evaluates individuals within a Manhattan distance of one. L9 evaluates individuals within Manhattan distances of one and two. C9 is a square-like neighbourhood which considers not only L5 individuals but also those at North-East, South-East, South-West and North-West. C13 is a combination of L9 and C9 local neighbourhoods.

Migration and selection intensity are directly related to cGAs performance. Here, both are evaluated separately; however a combination of both or a dynamic control of them is a possibility that need to be explored. In the next subsection, faults specification and their injection into data critical to the system are provided.

3.2.2 Faults Specification

The kind of faults the proposed cGA deals with are SHEs, but its inherent ability to overcome SEUs due to their temporary nature is assumed. In a maximization problem such as determining the attitude parameters of a vehicle; stuck at zero faults in fitness score registers imply low fitness values for faulty individuals. To simulate that effect, the fitness scores of faulty individuals are set to zero. Locally, the algorithm selects the fittest individual as the second parent for reproduction. Thus faulty neighbours are not selected and their solutions are not passed to the next generations. However, faulty individuals are possibly good solutions which are ignored, this consequence of a faulty scenario could induce an increase in convergence time and a possible loss in results accuracy. The stop condition of the cGA evaluates that all non faulty individuals converge to the global optimum.

The opposite case, stuck at one faults, is thought to be more critical. The local selection method chooses, recombine and spread the solutions of faulty individuals which might be poor solutions and, due to migration, possibly affect individuals located at far regions. In Subsection 3.2.4 is demonstrated that the fault tolerant algorithmic response deteriorates in this faulty scenario. Yet, the cGA still shows certain ability to converge.

The third faulty scenario evaluates the hybrid case. A combination of stuck at zero and one bits in fitness score registers which are injected with 50% probability of occurrence. In Subsection 3.2.4 the experimental constraints are described.

In Figure 3.4, an example of a worst case scenario with 50% faulty individuals is shown. In the example, the current evaluated individual migrates on a randomly determined direction (West) with a randomly determined distance (4 positions in Manhattan distance). Due to local selection, the central individual will always be the first parent for reproduction. After migration, the new neighbourhood might have less number of faulty individuals, as it occurs in the example. Several migration criteria are proposed based on the quality of solutions in the local neighbourhood. More details are presented in the next subsection.

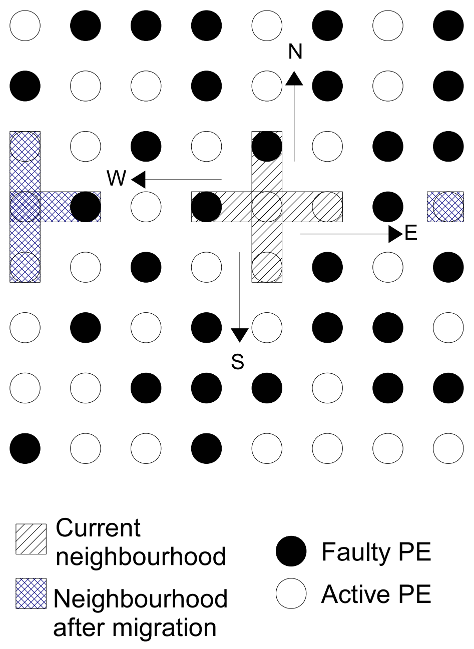


Figure 3.4: Worst case scenario, 50% faulty individuals

3.2.3 Algorithm Description

In [5], a double-stage cGA was proposed to tackle the GPS attitude determination problem. In the first or coarse stage, the algorithm aims to achieve a minimum 0.96 average fitness score, the maximum fitness value is close to 1.0 but less than 1.0 due to the noise effect. The second or fine stage is performed within a limit in the number of generations. During the coarse stage the search goes all over the search space. Once the minimum average fitness score is achieved and the algorithm goes into the fine stage, the search is locally performed on limited regions of the search space.

Migration among individuals is implemented as a mitigation technique to deal with faults at fitness score registers. In the cGA proposed here, a single search stage is performed with a higher convergence threshold of 0.98 average fitness score.

The algorithm promotes the central individual of a local neighbourhood to migrate to other neighbourhoods located in its own row or column (North, East, South and West directions). Implying a communications cost reduction because each individual is connected to those in its same row and column (rows + columns) instead of being connected to all individuals of the array (row * columns).

Migration distance and direction are randomly defined. Yet, three criteria for selecting recipient local neighbourhoods, are proposed: 1) migration occurs on a completely random base, 2) recipient local neighbourhoods (randomly selected in all directions) are evaluated in terms of their overall fitness score and selected accordingly and 3) individuals in recipient local neighbourhoods (randomly selected in all directions) are individually evaluated and selected to create a new neighbourhood.

On the other hand, the induced selection intensity is controlled through different local neighbourhood shapes and sizes, and in those cases only the effect of the local neighbourhood shape is evaluated without introducing any migration. These properties are evaluated separately to show their benefit but combinations of different migration rates and frequencies and local neighbourhoods configuration could benefit the performance of cGAs.

In Algorithm 5, the cGA pseudocode is presented considering a L5 local neighbourhood. For each individual the migration distance and direction are determined in the reproductive cycle. In line 11, the second parent is selected after migration is applied. For varying local neighbourhoods, lines 11 and 12 would change to individuals in corresponding neighbourhood configuration (L9, C9, C13), and lines 8 and 9 for migration are not executed.

Algorithm 5 Fault Tolerant cGA through Migration

```
1: procedure CGA
2:     GPS input data
3:     $(x) \leftarrow random\,(x_0)$                                   ▷ initial population
4:     $(f) \leftarrow evaluation\,(x)$                                 ▷ evaluation
5:     while $k \leftarrow 1, generations,$ or $\bar{f} \geq 0.98$ do
6:         for $i \leftarrow 1, popSize$ do
7:             if $migration$ then
8:                 $(ds, dr) \leftarrow random\,()$                     ▷ migration distance and direction
9:             end if
10:            $(f_{max}) \leftarrow max\left(f_{N_{ds,dr}}, f_{E_{ds,dr}}, f_{S_{ds,dr}}, f_{W_{ds,dr}}\right)$   ▷ second parent selection
11:            $(x'_1, x'_2) \leftarrow spc\,(x_i, x_{max})$            ▷ single point crossover
12:            $(x''_1, x''_2) \leftarrow mutation\,(x'_1, x'_2)$       ▷ offspring mutation
13:            $(f_{new}) \leftarrow max\,(f_i, f''_1, f''_2)$          ▷ offspring evaluation
14:            $(f_{temp}, x_{temp}) \leftarrow (f_{new}, x_{new})$     ▷ temporary population
15:        end for
16:        $(f, x) \leftarrow (f_{temp}, x_{temp})$                    ▷ whole population replacement
17:    end while
18: end procedure
```

3.2.4 Experimental Results and Analysis

Several test scenarios are considered in order to show the ability of the proposed fault tolerant cGA to overcome SHEs when migration is applied or selection intensity is controlled. The following experimental constraints are evaluated:

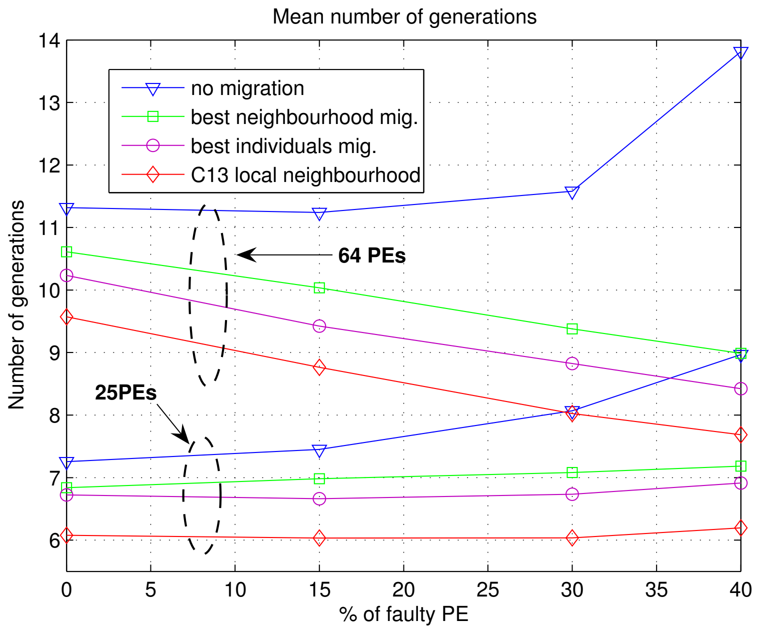
- Population size: 25, 36, 49 and 64 individuals
- L5, L9, C9 and C13 local neighbourhood configurations
- 10×10^3 independent runs per experimental case
- Maximum number of generations: 50
- The stop condition is the population's average fitness score where $\Delta\bar{f} > 0.98$

- The same random faulty distributions are applied in all cases
- The same input GPS data set is evaluated in all cases

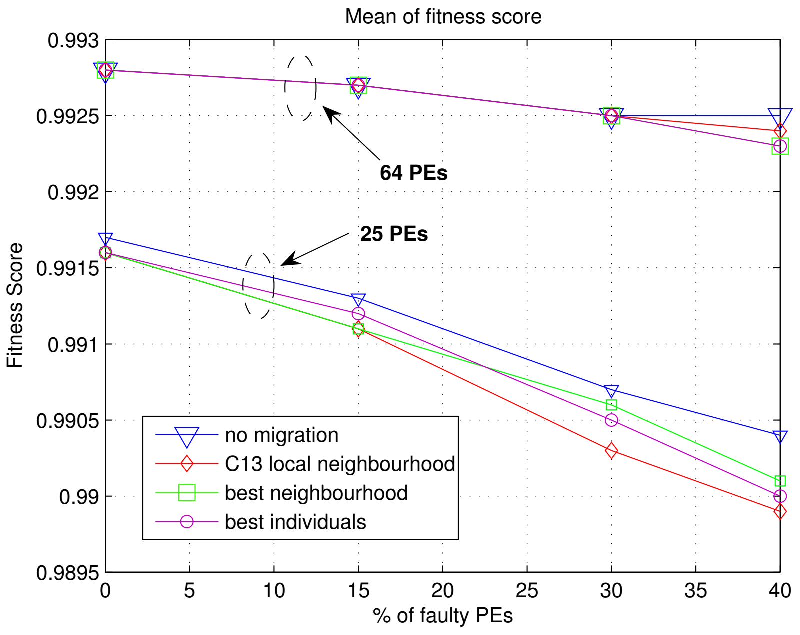
In Figure 3.5, results for 25 and 64 processor elements (PE) are shown considering stuck at zero SHEs at fitness score registers. Regarding convergence time, it is observed that the smaller the array size, the less time the algorithm requires to converge. However, when faults are induced and migration is applied or selection intensity is increased, the convergence time is reduced in both cases. For 64 individuals the effect of migration and selection intensity is noticeable because the number of generations reduces when the faults percentage increases. The reason is, for stuck at zero faults, the algorithm discards the solutions of faulty PEs and through migration or bigger local neighbourhoods the alive individuals sort out the faults and converge to the threshold. However, for hard real-time applications such as the one considered in this study, it is indispensable to maintain the results accuracy. In Figure 3.5(b), it is observed that an accuracy loss for 64 individuals is $6x10^{-4}$ while for 25 PEs, the loss is $2x10^{-3}$ in terms of fitness score. This loss of accuracy is the result of the population size plus the faults effect. Therefore, based on the results obtained, a population size of 64 individuals is more suitable for this application. Finally, in terms of the hit rate or the number of successful experiments; the bigger the array the better the hit rate when migration or selection intensity are applied.

The stuck at logic one case is more critical for the algorithm, due to the local selection method being based on the fittest individual and the consequent dissemination of poor solutions throughout the population. In Figure 3.7, the fitness score for alive PEs with faulty neighbours is presented. The worst cases correspond to those PEs which are surrounded by more than one faulty PE. If only one faulty PE is within the local neighbourhood the PEs can still converge to the threshold. To show the negative effect of this faulty scenario, the real fitness score of faulty individuals is obtained and included in the same graph. It is observed that poor solutions are spread through the array severely affecting not only adjacent but also distant located individuals due to neighbourhoods overlapping.

It is observed that without migration and the corresponding selection intensity of a L5 neighbourhood, the hit rate is unacceptable. Similar results are obtained for migration policies based on the fittest neighbourhood and the fittest neighbours



(a) Average number of generations for successful experiments.



(b) Population's average fitness score.

Figure 3.5: Population sizes of 25 and 64 PEs with stuck at zero SHEs at fitness score registers. Fault mitigation techniques are migration and the configuration of the local neighbourhood.

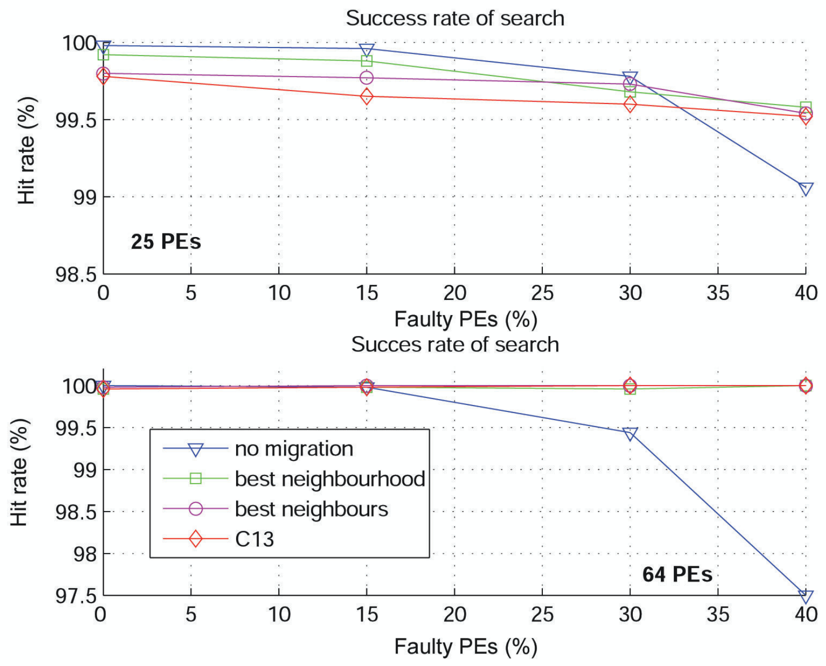


Figure 3.6: Hit rate expressed as the percentage of successful experiments from the total number of independent runs.

criteria. However, with a random migration policy, the algorithm converges with up to 30% faulty PEs for all array sizes with a considerable increase in the convergence time while maintaining the results accuracy. In contrast, the hit rate is affected due to the spread of poor solutions through the array.

Finally, the hybrid case for stuck at logic zero and one at phenotypes is evaluated. The same experimental constraints are assessed considering 50% probability for zero or one bit flips to occur. After applying the same migrations policies and different local neighbourhoods, the cellular GA has shown its ability to deal with faults when applying a random migration criterion. For other migration policies and local neighbourhood shapes, the algorithm fails in finding the global optimum. In Figure 3.8 it is shown how the convergence time increases up to 30 generations in the worst case, while better accuracy is achieved through a larger population size.

On the other hand, there is a significant loss in the hit rate for 64 individuals. The reason is the increase in the number of faulty individuals affected by the spread of poor solutions, with stuck at one faults, throughout the array. Although the percentages of faults for all array sizes are the same, the faults impact is different. For example, with 15% of faulty PEs, a 25 array deals with up to 4 faulty individuals, while 64

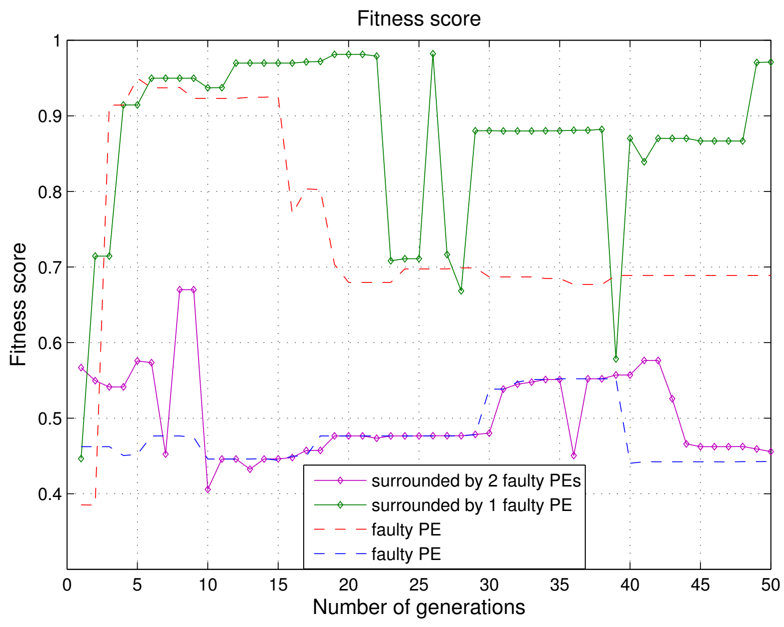


Figure 3.7: Non-faulty individuals' fitness scores surrounded by one or more faulty individuals with stuck at one SHEs in fitness score registers and corresponding faulty individuals' fitness scores are shown for one experimental sample.

individuals deals with up to 10 faulty PEs. Thus, the number of affected individuals every generation is proportional to the population size. In the next subsection, a summary of this section is presented.

3.2.5 Summary

The ability of cellular GAs to deal with SHE, and consequently with temporary SEU, has been empirically demonstrated through the integration of migration and controlled selection intensity, both specific to cellular GAs. Several migration policies were tested based on different selection and replacement policies; moreover, the selection intensity has been modified through changing the local neighbourhood shape and size. It has been shown that fitness score registers are critical for cGAs based architectures. Faulty fitness registers misguide the search spreading poor solutions and in worst case scenario the search stagnates or converge to a local optima. However, it has been empirically demonstrated that the algorithm could deal with stuck at zero faults without affecting the performance and reliability of the system. Instead, in the presence of stuck at one faults, the algorithm can converge only when a random migration policy is applied and

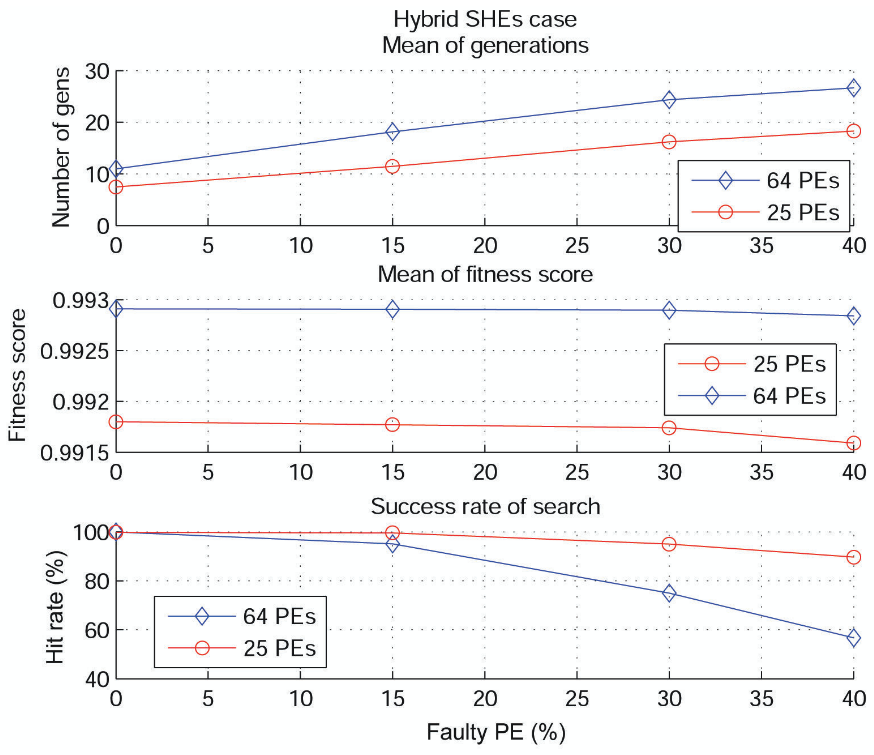


Figure 3.8: Hybrid SHEs (stuck at zero and one) at fitness score registers. Top graph: average number of generations for successful experiments. Amid graph: Population's average fitness score for successful experiments. Bottom graph: Hit rate or number of successful experiments.

faults affect up to %30 of PEs. Similar observations were true for the hybrid case, stuck at logic one and zero bits. However in this case the cellular GA could deal with larger faults percentages. Migration and different local neighbourhood configurations are characteristics specific to cGAs. Boundary conditions for migration and several local neighbourhoods configurations have been considered and explored, in order to obtain a general overview of cGAs abilities to deal with faults at algorithmic level. However, designing efficient, accurate and fault tolerant cGAs implies to finely tune these and other parameters, not only at topological level but also at algorithmic level. In the next section, the genotypic space is subjected to SEEs and a fault tolerant algorithmic perspective is provided.

3.3 Faults At Genotypes

In this section, a fault tolerant approach is developed for the genotypic also known as the representation space of GAs. Individuals or chromosomes are normally characterized through binary or real strings, though other representations are also possible. In this thesis, binary encoding has been used to solve the GPS attitude determination problem. Yet, problem representation in the evolutionary arena is a research line by itself, for more details the reader is referred to [7].

The same criterion of Section 3.2 is followed. To investigate the inherent ability of cGAs to deal with SHEs that could permanently affect the operation of a system. SEUs are implicitly evaluated due to their temporary nature. The proposed approach is based on detecting significant changes in genotypic diversity during the evolutionary process and thus to control the exploration-exploitation trade off through modifying the neighbourhood to grid ratio (NGR), a parameter specific to cGAs, see Subsection 2.4.1 for more details. By appropriately controlling the NGR, the difficult search space associated to the attitude determination problem is conveniently explored in terms of efficiency and efficacy.

In this respect, permanent SHEs and temporary SEUs are targeted at functional level through the dynamic adaptation of the topology of cGAs. Adding to the fault tolerant cGA presented in the previous section now to deal with faults at the genotypic space. Here the algorithmic approach aims at quantifying the effect of SHEs in chromosomes registers by measuring the loss of genetic diversity and thus to react by changing the lattice configuration to mitigate the faults effect. The proposed algorithmic description is introduced in the next subsection.

3.3.1 Algorithm Description

Structurally, cGAs possess several characteristics that can directly affect the evolutionary process and therefore the search's outcome. As mentioned before, among those features are the lattice and local neighbourhood size and shape, the implemented local selection methods, the implicit diffuse migration due to neighbourhoods overlapping, among others. De Jong et al. assigned an unique numerical measure to the effect of combining different lattice and local neighbourhood sizes and shapes, that parameter is known as the neighbourhood to grid ratio (NGR) [31]. In Figure 2.4, the NGR is drawn for different sizes and shapes of lattices with a L5 local neighbourhood. Corresponding take over times are shown in Figure 2.7 for square, rectangular and narrow lattices also with a L5 neighbourhood considering a population of 400 individuals. Results demonstrated that having lower NGRs indicates more explorative search while higher NGRs perform a more aggressive search.

Recently, dynamically changing the lattice shape at run time has been investigated as a resource for appropriately balancing explorative and exploitative stages of the search [6]. A variety of problems presenting different levels of difficulty were tackled in order to demonstrate the approach. Promoting solutions exploration or exploitation through switching the topology configuration improves the search quality in terms of convergence time and results accuracy. An adaptive cGA implementing a speed measure for the loss of diversity at the genotypic and phenotypic spaces is proposed. Results showed improved performances through the proposed approach.

Investigating dynamic changes in the population topology from a fault tolerant perspective at the genotypic space is the main aim of the proposed algorithmic approach. Measuring the loss of genetic diversity due to SHEs -implicitly SEUs- at chromosomes registers in order to induce a reconfiguration step in the population topology of the cGA is suggested. In Figure 3.9 an example of a common faulty scenario is shown. Considering that one or more bits in chromosome x_0 are faulty, stuck at one or zero, and possibly one or more chromosomes of its neighbourhood contains faulty bits too, in the example x_s is also faulty. After local selection and recombination, and even though x_0 mates with a non faulty individual, x_e, faulty bits are inherited by the offspring, altering their current genetic material and possibly affecting individuals quality for future generations. In a way, SHEs/SEUs at chromosomes registers are a form of mutation, with no rules attached. Due to the permanent nature of SHEs, the effect of faulty bits will remain in the upcoming generations, while SEUs will be overcome due to their temporary nature. It is also important to remember that, mutation has in itself a great effect in cGAs behaviour, as showed in [76, 77].

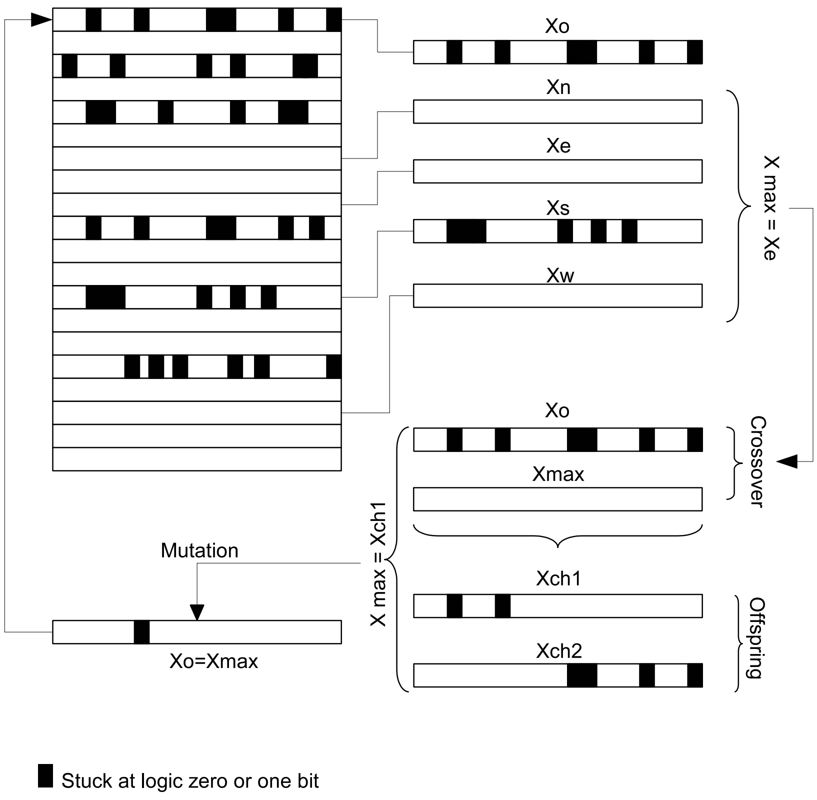


Figure 3.9: Faulty scenario

Genetic diversity is lost as a consequence of SHEs or SEUs. Measuring the genetic diversity loss during the search provides enough information to dynamically commute between square and rectangular grid shapes and promote exploitation or exploration as required. Balancing exploitation and exploration during the search is necessary. Excessive exploitation would result in premature convergence to a local optimum. On the other hand, too much exploration would guide the algorithm to areas of poor solutions in the search space. Consequently, the processing time would increase and in the worst case scenario the search would stagnate. If faults are added to this scenario and SEEs affect the chromosomes registers, to maintain an adequate balance becomes more important, because non faulty individuals would be able to sort out faulty ones and evolve towards the global optimum.

Genetic diversity is calculated based on individuals entropy which is given by the Hamming distance among chromosomes from one generation to another. In order to overcome a quick loss of genetic diversity due to the faults, exploration must be promoted through migration. Thus, individuals surrounded by faulty ones could mate with distant and possibly non-faulty individuals.

In Algorithm 6 shows the cellular GAs flow chart including the adaptive criteria to dynamically switch between topologies. $\Delta H_t = H_t - H_{t-1}$ is the average Hamming distance between two continuous generations. A decrease in that difference by a factor (ϵ) implies slow evolution and consequently excessive exploration [6]. Exploring in excess would increase the number of generations required to converge to the global optimum, because the cGA would spend valuable time searching in areas of the landscape with poor solutions. Thus, switching to a topology with higher NGR is desirable. On the other hand, if the average Hamming distance between two consecutive generations increases by a normalized difference of the same factor ($(1 - \epsilon)$), it means the search process is going fast, and therefore too much exploitation is being applied and there is high probability of premature convergence. Promoting exploration by switching to a topology with lower NGR would improve the search process. If none of those conditions are fulfilled the algorithm evolves using the last configuration of the topology.

This adaptive approach was introduced in [6] with an $\epsilon = 0.05$ factor experimentally determined as an optimum value, based on the results obtained from a set of problems comprising several optimization features such as: epistasis, multi-modality, deceptiveness, etc. The main advantage of this approach is its inexpensive computational cost which is $O(p \cdot l)$ where p corresponds to the number of individuals and l to chromosomes length. In the next subsection, the experimental results and their analysis are presented.

3.3.2 Experimental Results and Analysis

Several test scenarios are considered in order to show the ability of cGAs to overcome SHEs/SEUs at chromosomes registers by controlling the NGR. The following experimental constraints are assessed:

- Array sizes of 25, 36, 49 and 64 individuals for square topologies, and 24, 36, 48 and 64 for rectangular topologies,

Algorithm 6 Adaptive fault tolerant cGA based on genotypic diversity

```
1: procedure CGA
2:     GPS input data
3:     $(x) \leftarrow random\,(x_0)$                                   ▷ initial population
4:     $(f) \leftarrow evaluation\,(x)$                                 ▷ evaluation
5:     while $k \leftarrow 1, generations,$   or   $\bar{f} \geq 0.98$ do
6:         for $i \leftarrow 1, popSize$ do
7:             $(f') \leftarrow selection\,(f_N, f_E, f_S, f_W)$            ▷ second parent selection
8:             $(x'_1, x'_2) \leftarrow recombination\,(x_0, x')$           ▷ parents recombination
9:             $(x''_1, x''_2) \leftarrow mutation\,(x'_1, x'_2)$            ▷ offspring mutation
10:            $(f_{new}) \leftarrow evaluation\,(x_0, x''_1, x''_2)$         ▷ evaluation
11:            $(f_{temp}, x_{temp}) \leftarrow replacement\,(f_{new}, x_{new})$   ▷ temporary allocation
12:        end for
13:        $(f, x) \leftarrow (f_{temp}, x_{temp})$                        ▷ synchronous population's updating
14:        if $\Delta H_t - \Delta H_{t-1} < \epsilon \Delta H_{t-1}$ then          ▷ promote exploitation
15:            Evolve on a square grid;                                  ▷ high NGR
16:        else if $\Delta H_t - \Delta H_{t-1} > (1-\epsilon)\,\Delta H_{t-1}$ then   ▷ promote exploration
17:            Evolve on a rectangular grid;                             ▷ low NGR
18:        else
19:            Evolve on previous generation grid's shape;
20:        end if
21:    end while
22: end procedure
```

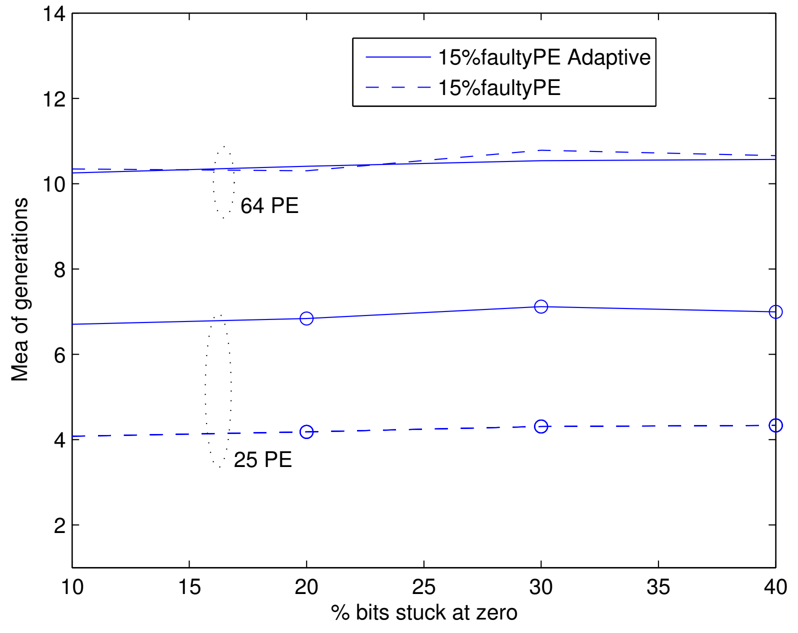
- L5 local neighbourhood,
- $5x10^3$ independent runs were carried out per experimental case,
- A maximum of 50 generations,
- The same random faults distributions are applied in all experimental cases,
- The same input GPS data set is evaluated,
- Only non faulty individuals are considered for threshold evaluation.

Due to the large number of experimental samples, only results corresponding to 25 and 64 individuals are included in this section. Cellular GAs performances for intermediate population sizes are observed to be in between those results reported for the smaller and the larger population sizes. The efficiency of the algorithm is measured in terms of convergence time as the number of generations required to obtain an optimal solution. The algorithmic efficacy is evaluated considering the hit rate together with the results accuracy.

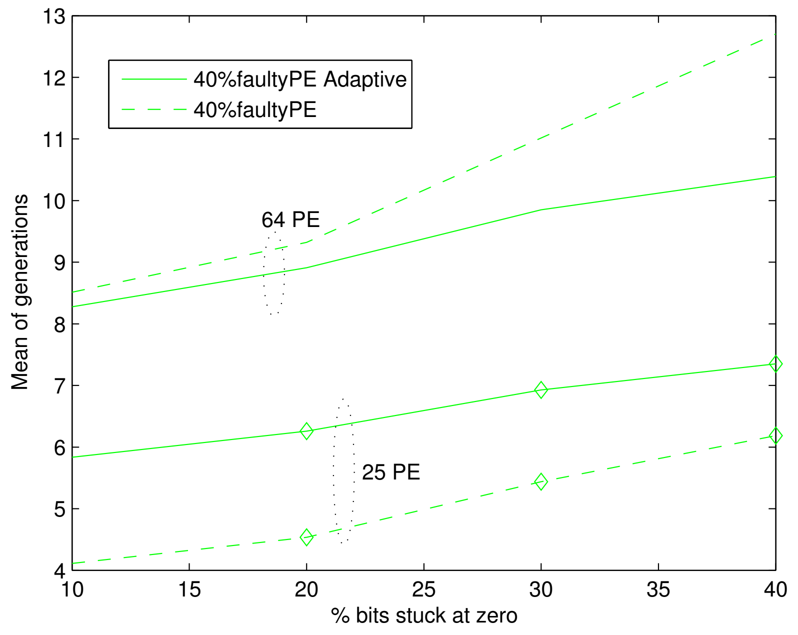
3.3.2.1 Stuck At Zero Faults

In Figure 3.10, results are presented considering stuck at zero SHEs at chromosomes registers. In terms of convergence time, for lower percentages of faulty chromosomes (15%), 64 PEs behave in a similar way in both normal and adaptive approaches. In contrast, the smaller array 25 individuals, increases its convergence time with the adaptive approach. With high fault percentages (40%), the adaptive approach shows a reduction in convergence time for 64 individuals, meanwhile it increases for an smaller array. In that case, genetic diversity provided by a large number of individuals improves the efficiency of the larger array.

The results accuracy of 25 PEs shows the adaptive approach outperforms the standard cellular GA. In Figure 3.11(a), in all percentages of faulty individuals, higher average fitness scores were achieved. On the other hand, for 64 PEs, the adaptive approach improvement is noticeable for 30% and 40% faulty individuals. However, experiments show better results accuracy with the smaller array. In terms of efficacy, the achieved hit rate is 99% through the adaptive approach while 95% is obtained by a standard cGA.

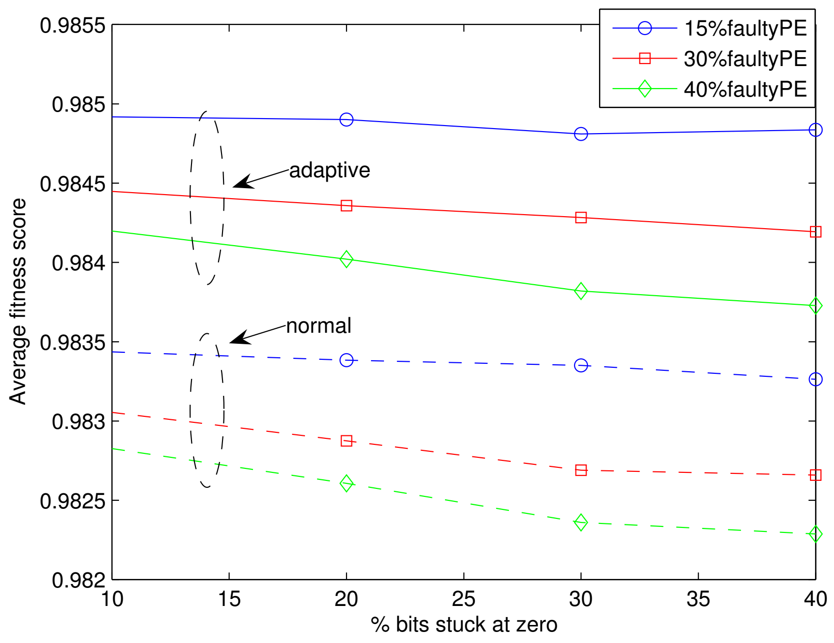


(a) Average number of generations

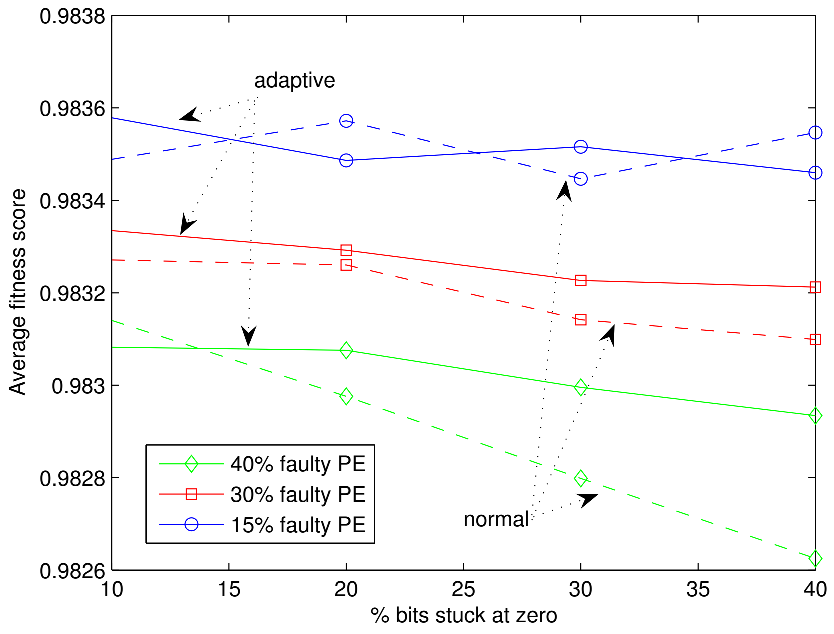


(b) Average number of generations

Figure 3.10: Faulty Chromosomes, SHEs Stuck at zero



(a) 25 PEs - average fitness score



(b) 64 PEs - average fitness score

Figure 3.11: Faulty chromosomes, average fitness - SHEs Stuck at zero

3.3.2.2 Stuck At One Faults

Regarding stuck at one SHEs at chromosomes registers, positive results were also obtained. For 25 PEs, the convergence time is reduced when adaptive approach is applied with fault percentages of 30% and 40%; while the average fitness score is superior in all cases when compared to the performance of a standard cGA. 64 individuals show an overall reduction in convergence time for all percentages of faults, while the results accuracy is also improved.

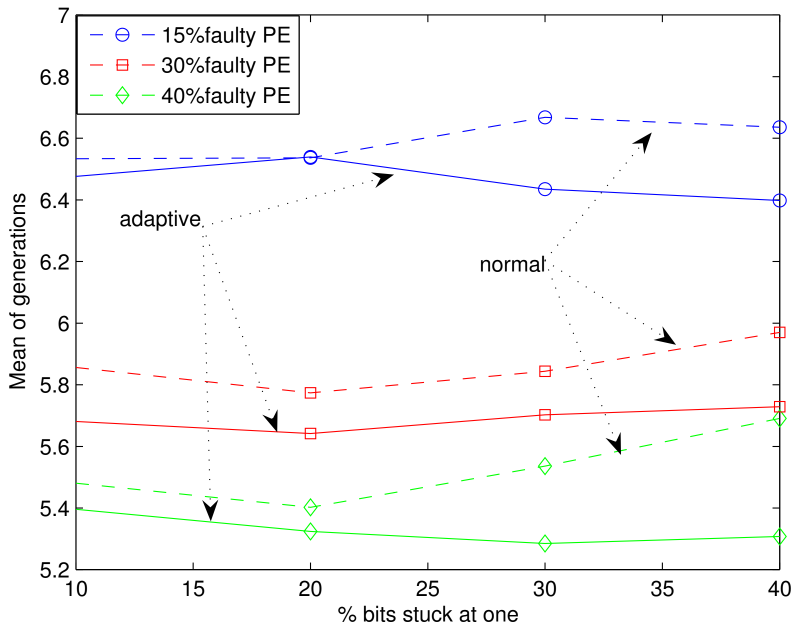
Chromosomes with faulty bits stuck at one imply in the worst case scenario that corresponding faulty variables would permanently or temporarily represent their maximum rank values (the opposite occurs in the stuck at zero case). However because of the adaptability of the algorithm, faulty chromosomes are sorted out when migration is induced by the reconfiguration of the topology. The hit rate is maintained in 99% for the adaptive and the standard cellular GA.

3.3.2.3 Hybrid Stuck At Faults

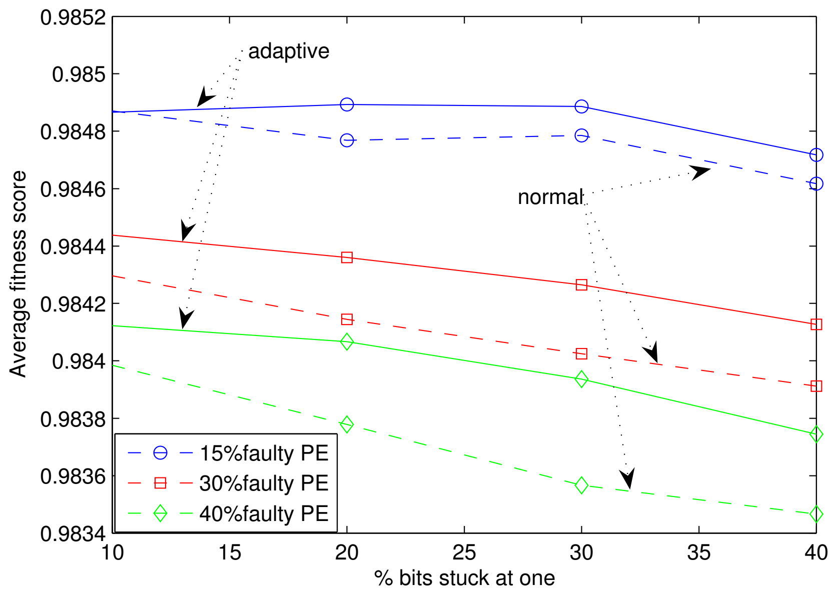
Finally, random hybrid faults distributions (stuck at one or zero) are injected in chromosomes registers. The same experimental constraints are assessed considering equal probability for stuck at logic zero or one bit flips to occur. Results showed an intermediate performance between the two faulty cases evaluated before (stuck at zero or at one faults). In terms of the average number of generations, the proposed adaptive approach does not improve the convergence time. Having stuck at zero and one bits in chromosomes registers could affect individuals fitness scores in many different ways. For example, if significant bits per encoded variable are affected the individual fitness score can erroneously and dramatically change. On the other hand, if less significant bits per encoded variable are affected, it may still be possible for those faulty individuals to spread a solutions that is not importantly affected by the faults. On the other hand, results accuracy for the standard cGA is improved by the adaptive approach. The hit rate is the same than in the previous stuck at one faulty scenario.

3.3.3 Summary

In this section, the ability of cGAs to deal with SHEs affecting chromosomes registers was empirically demonstrated. An adaptive approach to change the topology configuration while measuring the lost of genotypic diversity due to the faults was proposed

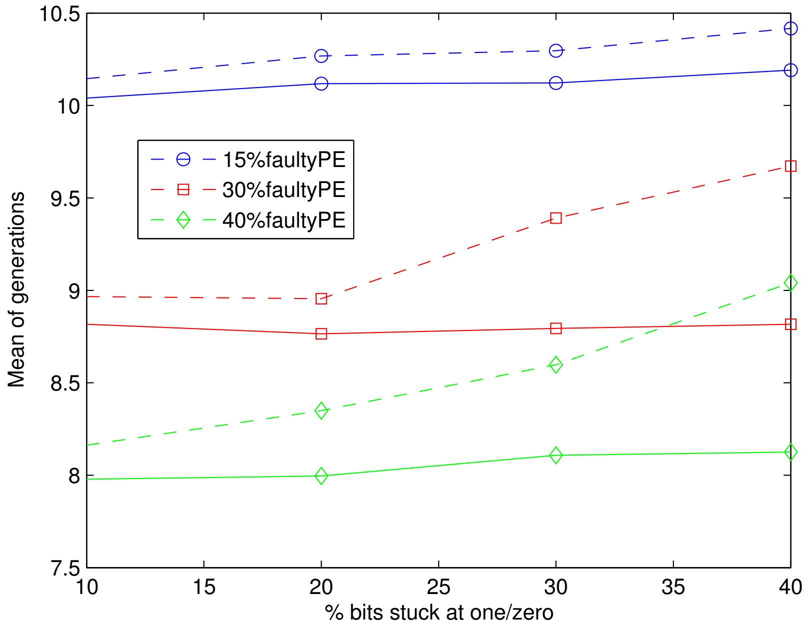


(a) Average number of generations

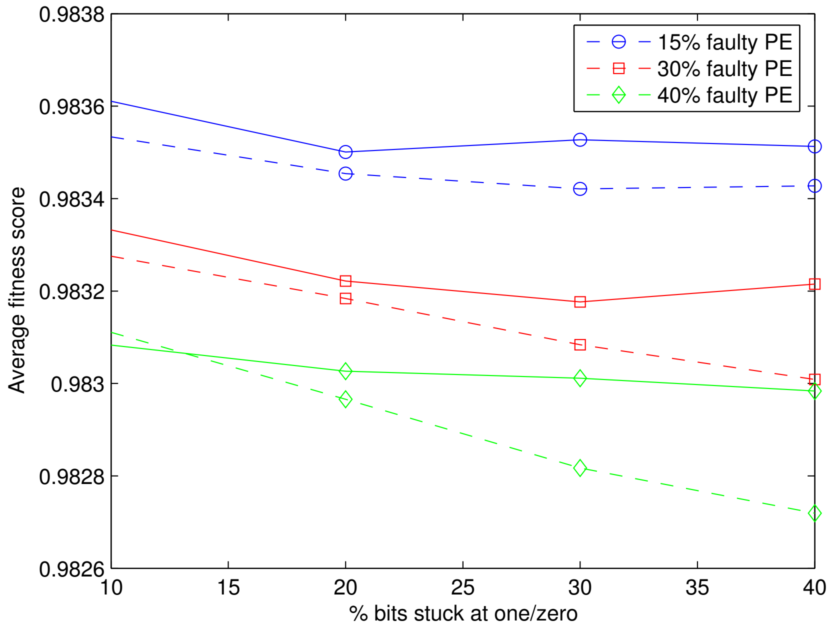


(b) Average fitness score

Figure 3.12: Faulty chromosomes, 25 PEs, SHEs stuck at one



(a) Average number of generations



(b) Average fitness score

Figure 3.13: Faulty chromosomes, 64 PEs, SHE stuck at one

and assessed. Changing the topology shape provides either a more explorative (low NGR) or more exploitative (high NGR) search. The main objective of this research is to propose mitigation techniques that take advantage of the inherent properties cGAs possess to deal with permanent bit flips affecting chromosomes data. This study does not pursue a comparison with today's state of the art GAs or any other non-evolutionary optimization tool but to further investigate and extend to the fault tolerant arena the implicit capabilities of cellular GAs.

The experimental results showed a superior performance for the adaptive approach regarding results accuracy. Convergence time was better maintained when large array sizes were used. The hit rate was similarly maintained in both approaches with minimum advantage for the adaptive approach for stuck at zero faults.

Measuring genotypic diversity and adaptively commute to a lower or higher NGR has been an initial step in the investigation of the inherent abilities of cGAs to deal with faulty scenarios. Monitoring not only genetic diversity but also phenotypic diversity changes during the evolutionary process, is a desirable combination from a fault tolerance perspective. Although genetic changes normally imply phenotypic changes, the opposite consideration is not always valid. Phenotypic changes do not always represent an alteration in the genotypic space, and fault tolerance is an example of that. For example, if faulty chromosome registers have a high fitness score due to stuck at one faults, and therefore higher fitness scores are obtained, this does not necessarily mean the algorithm is approaching a global optimum. On the contrary, obtaining higher fitness scores due to genetic diversity loss could cause the algorithm to stay trapped at a local optimum. Thus, more restrictive criteria for algorithm adaptability are needed. In the next section, a distributed approach is first analysed aiming to improve the results accuracy without adding computational cost. Thereafter, a fault tolerant approach is incorporated. Several experimental set-ups are implemented in order to validate the proposed approach while the same case study of this section is tackled.

3.4 A Distributed Approach

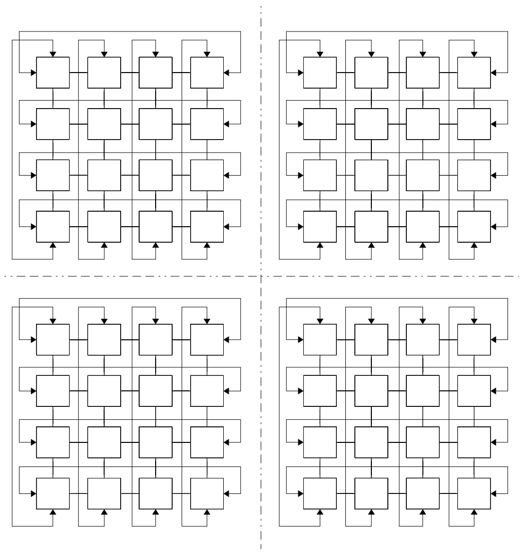
In this section, a distributed cellular Genetic Algorithm (dcGA) for the implementation of the GPS attitude determination system is proposed and tested. Previously, a cellular GA architecture was proposed and comparison among different standard and parallel implementations was carried out. However, comparison among those reveals that accuracy is compromised when the population size is increased. In this section, a distributed configuration approach is proposed and compared with previous implementations [5]; a significant improvement in terms of accuracy is reported without increasing the computational cost.

A fine cellular parallel GA was presented in [51], targeting the GPS attitude determination problem. Results from a set of experiments with different parameters configuration showed that cellular GAs outperform sequential ones in terms of efficiency and efficacy. Moreover, another set of experiments considering cellular configuration and random migration for individuals was also evaluated. It was noted that when population size increases the average fitness score drops significantly. Similar behaviour was observed for sequential and parallel GA implementations. Meanwhile, the average number of generations decreased as expected and the hit rate was maintained. The combined approach introduced here aims at improving results accuracy without increasing computational resources of previous implementations. In order to achieve this, a combined distributed and cellular GA is proposed. Initially, a fixed configuration is implemented with only certain individuals mating others from different sub-populations while evolving in parallel.

In Section 2.3, distributed PGAs are introduced including an analysis of their main characteristics. In [42] and [43], distributed PGAs were tackled through a combined coarse and fine GA evolving in parallel with predefined time slots for interaction among sub-populations. It was identified that excessive interaction among sub-populations could lead the algorithm to perform worse than panmictic approaches. Therefore, defining an adequate migration criteria is one of the main issues in distributed PGAs. On the other hand, Gordon et al. evaluated a multiprocessor cGA implementation, subdividing the whole population in small sub-populations with predefined intervals for interaction among them [78]. Contrary to expectations, this island approach performed poorly in comparison to a fully connected grid structure for a test bench of difficult optimization functions. As mentioned before, a difficult optimization task such as the GPS attitude problem is targeted here using a distributed cellular GA approach, subdividing the entire population into sub-populations with information exchange among these only through predefined individuals. Results show to improve a standard cellular GA.

3.4.1 Algorithmic Approach

The cellular GA proposed here is a combination of a distributed and a cellular parallel GA. A structured population on a square grid is divided into sub-populations. In Figure 3.14 the overall connection scheme is presented. Each sub-population is connected according to a toroidal like shape with its edges wraparound, thus each sub-population evolves independently. Interaction among sub-populations is performed through individuals located at sub-populations corners. Those individuals change their local neighbourhoods configuration to consider individuals located at adjacent sub-populations corners (Figure 3.14(b) red lines) for selection, and in the case those were stronger individuals in comparison to local ones, for reproduction. Individuals located at grid corners require a total of 2 remote accesses (during migration events) and 4 local accesses for selection and replacement. In this study, interaction among sub-populations is defined by two migration policies: 1) maximum migration frequency and 2) an adaptive migration criterion have been defined based on the loss of phenotypic and genotypic diversity during the evolutionary process.


(a) Isolated cellular populations

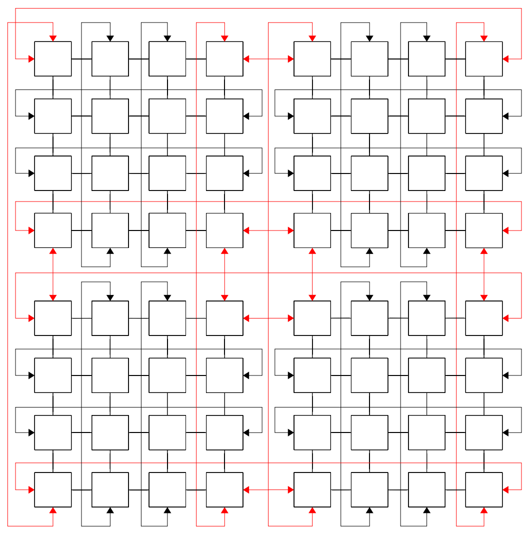

(b) Individuals at corners reproduce among sub-populations

Figure 3.14: Algorithm connection diagram

In [20] and [3] an extensive study of main aspects in PGAs and particularly migration policies are covered. Random and best individuals selection and replacement criteria showed to improve the search quality and to reduce the convergence time.

3.4.1.1 Adaptive Migration

Defining migration policies is one of the main issues in parallel GAs. Applying the maximum migration frequency (migration takes place every generation) is commonly used as a reference point for comparison. However, it is known that applying high migration frequency could lead the search to an exploration excess, where individuals would probably spend time looking at areas of the landscape with poor solutions. Here, the adaptive approach measures the loss of phenotypic or genotypic diversity inherent to the evolutionary process and allows predefined individuals to migrate for reproduction.

Algorithm 7 shows the distributed cellular GA pseudocode for a square grid population subdivided into four sub-populations, considering a L5 local neighbourhood defined by individuals at North, East, West and South of each chromosome position. Diversity at phenotypic and genotypic spaces is evaluated: 1) measuring the average fitness score difference of current and previous generations shows the search progress in the phenotypic space; 2) the population entropy is measured using the Hamming distance between individuals from one generation to the other, providing algorithm behaviour in the genotype space.

Changes in the phenotypic space are defined by $\Delta \bar{f}_t = \bar{f}_t - \bar{f}_{t-1}$ and $\Delta \bar{f}_{t-1} = \bar{f}_{t-1} - \bar{f}_{t-2}$, which are differences between the current and previous generations in terms of the average fitness score. On the other hand, genotypic variations are calculated by $\Delta \bar{H}_t = \bar{H}_t - \bar{H}_{t-1}$ and $\Delta \bar{H}_{t-1} = \bar{H}_{t-1} - \bar{H}_{t-2}$, as introduced in [6]. Both conditions are implemented separately and results are presented in the next subsection. In both scenarios, increments in more than the normalized difference of a factor $(1 - \epsilon)$ in the phenotypic or genotypic measures, mean that the search process is going fast, with the risk of a quick loss of diversity and a consequent stagnation in the search. Thus, exploration is pursued through sub-populations jointly evolving and exchanging genetic material through their corners. However, if that difference decreases in less than a factor (ϵ), exploitation should be promoted, through independent evolution of sub-populations. The *epsilon* factor is the same used previously in Section 3.3. Having different values of this parameter imply a stronger or a weaker threshold to determine the speed of evolution.

An advantage of this dynamic control of interaction among sub-populations is its low computational cost. Complexity to calculate the average fitness of a population of n individuals is $O(n)$ and for population's entropy is $O(n \cdot l)$, where l is the chromosome

Algorithm 7 Distributed cellular GA algorithm

```
1: procedure dcGA
2:     $(x_1, x_2, x_3, x_4) \leftarrow random\,(x_0)$                     ▷ initialize sub-populations
3:     $(f_1, f_2, f_3, f_4) \leftarrow evaluation\,(x_1, x_2, x_3, x_4)$     ▷ evaluation
4:     while $k \leftarrow 1, generations,$ or $\bar{f} <= threshold$ do
5:         if $\Delta\bar{f}_t - \Delta\bar{f}_{t-1} > (1-\epsilon)\,\Delta\bar{f}_{t-1}$ then     ▷ sub-populations evolve jointly
6:         else                                                ▷ sub-population evolve independently
7:         end if
8:         for $i \leftarrow 1, pop\_size$ do
9:             $(f', x') \leftarrow selection\,(f, x)$                  ▷ parents selection
10:            $(x'') \leftarrow recombination\,(x')$                  ▷ parents recombination
11:            $(x''') \leftarrow mutation\,(x'')$                     ▷ offspring mutation
12:            $(f_{new}) \leftarrow evaluation\,(x''')$               ▷ offspring evaluation
13:            $(f_{temp}, x_{temp}) \leftarrow (f_{new}, x_{new})$     ▷ temporary storage
14:        end for
15:        $(f, x) \leftarrow (f_{temp}, x_{temp})$                      ▷ replacement
16:    end while
17: end procedure
```

length. In the following section, experimental constraints and results are presented.

3.4.2 Results Analysis

In [51], Xu et al. proposed a cellular GA to tackle the GPS attitude determination problem. A comparison among different population sizes and algorithm configurations was carried out. Initially, a set of standard GA configurations was implemented and compared. A cellular GA with a L5 local neighbourhood (formed by North, East, West and South (NEWS) individuals) and two search stages was introduced. In the first or coarse stage the aim was to achieve a population's average fitness score of 0.96 and then a second or fine stage was performed based on the results obtained in the coarse stage; the evolutionary process is stopped within 200 generations of the fine stage [51]. On the other hand, two versions of a sequential GA were also implemented and compared: one with a single search stage and a second with two, coarse and fine, search stages. Results showed a superior performance of the cellular GA in terms of efficiency (number of generations) and efficacy (accuracy and hit rate). A second set of experiments was also evaluated; the main experimental difference was the introduction of a probability based migration criterion. Migration occurs when an individual evaluates and mates other from a random or a predefined neighbourhood located at a certain distance. In that set of experiments, similar performance was obtained in terms of the mean average fitness score and hit rate, as well as with the average number of generations. The results accuracy in both experimental sets was lost when the population size was increased [5, 51]. High accuracy is particularly essential for the GPS attitude application tackled in this study. An increase in the threshold of the stop condition, which is the population's average fitness score, from $\bar{f} = 0.98$ originally used in [5, 51] to $\bar{f} = 0.99$ implies a variation in the outcome of the attitude parameters of approximately (due to the noise) $\varphi = 0.5$ degrees in the azimuth angle and $\beta = 0.1$ degrees in the elevation angle. The experimental results for the algorithmic approach presented here, show improvement in the results accuracy without increasing the algorithmic cost.

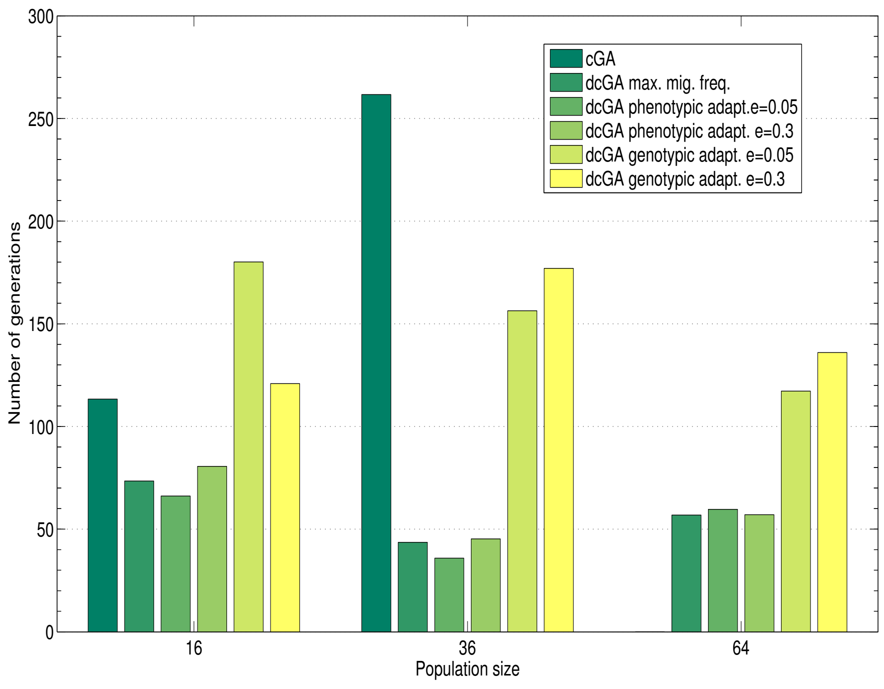
The next experimental constraints are defined:

- Population sizes of 16, 36 and 64 with sub-population sizes of 4, 9 and 16 individuals,
- L5 local neighbourhood with five individuals in NEWS shape,

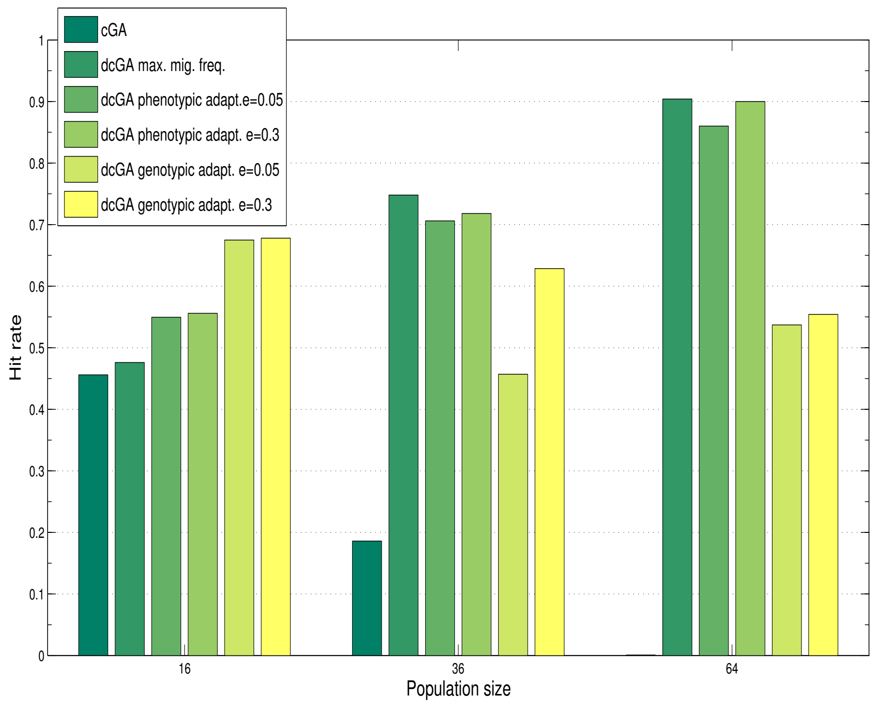
- A limit of 500 number of generations,
- The stop condition is the population's average fitness score where $\Delta\bar{f} > 0.99$.
- The same GPS input data set is evaluated in all experimental cases,
- 500 independent runs are performed per case.

As explained in section 3.4.1, the proposed distributed cellular GA subdivides the entire population into four sub-populations; as a reference, internal sub-population data accesses are considered as local and among sub-populations as remote, regardless the actual implementation platform. Interaction among sub-population is only allowed for individuals at corners. For reproduction, each individual at interior grid positions, requires 6 local data accesses; 4 for selection (4 neighbours), 1 for individual content retrieval and 1 for offspring placement. Meanwhile, corner individuals perform 4 local and 2 remote data accesses. If $\sqrt{n_{sub}}$ is the sub-population side size, and only individuals at corner can migrate, in total $4\left(\sqrt{n_{sub}} - 2\right)$ remote accesses per sub-population are avoided.

In Figure 3.15(a), results for the average number of generations are presented. For 16 and 36 individuals, the standard cellular GA increases the convergence time to achieve the average fitness score threshold. However, for 64 individuals the algorithm cannot converge to the threshold in any of the experiments. In contrast, when the distributed cellular configuration is applied, the convergence time is improved in all array sizes. The performance difference in terms of convergence time among migration policies shows a slight improvement through phenotypic adaptive migration in comparison to the maximum migration frequency; the opposite happens with genotypic based migration. In terms of the hit rate, see Figure 3.15(b), the largest array outperforms smaller populations achieving up to 90% success. However, even for smaller arrays, the hit rate is improved through the distributed cellular approach, except for the genotypic based migration policy which performs poorly. The performance of a population size of 64 individuals highlights the improvement of having a parallel cellular GA. In comparison to previous results [51], where the largest population size of 49 individuals on a static cellular structure provided an accuracy of 0.980 of the average fitness score. Results obtained in this study show that the distributed cellular approach overcomes this threshold without increasing processing resources or algorithmic steps.



(a) Average number of generations



(b) Hit rate

Figure 3.15: Results bar charts

In Figure 3.16, a random experimental sample is drawn showing some snapshots during the evolutionary process: at the beginning, in the middle and at the end of an average successful experiment. Darker squares correspond to highest fitness scores and vice versa. Considering a population size of 64 individuals, the same random initial population is used in all samples. In the middle of the search, the distributed cGA has almost conquered the whole population with individuals achieving at least the average fitness score. Instead, the standard cGA shows more individuals with lower fitness scores and therefore more phenotypic diversity. Yet, those observations are not conclusive, because of the stochastic nature of genetic operations where higher phenotypic diversity does not necessarily means the search would stagnate or would converge to a local optimum. However, final snapshots reveal something interesting, in standard cGAs, phenotypic diversity is regained, and due to the applied stop condition several individuals cannot converge. Distributed cGAs follow a better search strategy by having more exploitative sub-populations and higher hit rates are in general achieved. In the next subsection a summary of this section is presented.

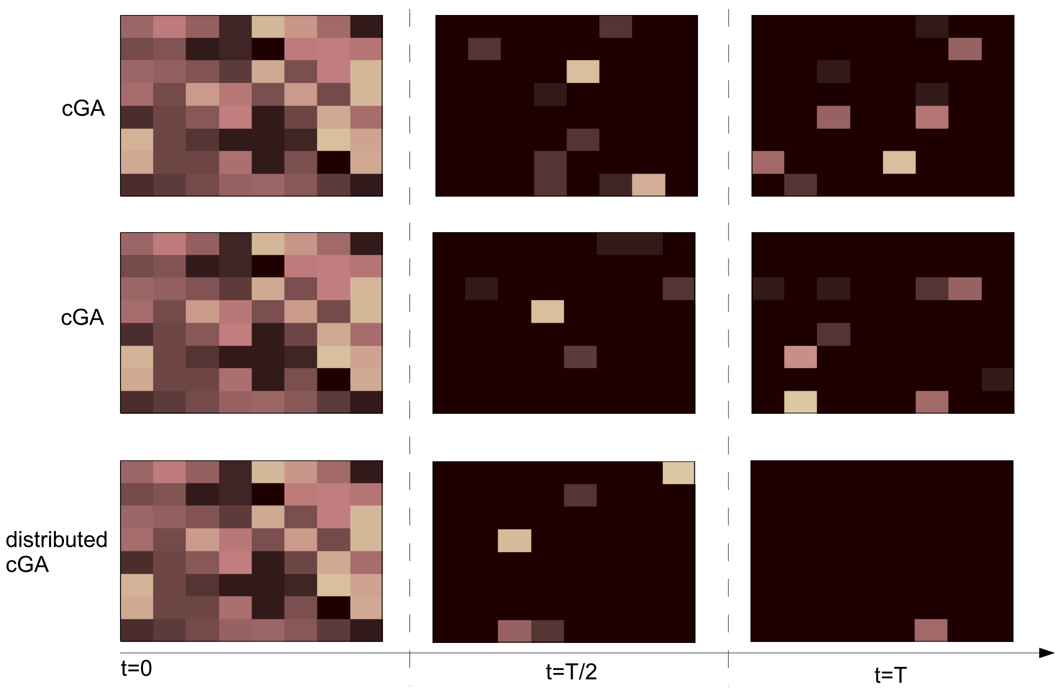


Figure 3.16: Snapshot Samples - cellular GA vs distribute cGA

3.4.3 Summary

A combined distributed and cellular GA has been compared with a standard cellular GA. The case study was the GPS attitude determination problem which presents a difficult search space with multiple peaks. Improving the accuracy of the attitude parameters has been the main aim in this study. Therefore a high convergence threshold of 0.99 of the average fitness score has been used in all experiments. The distributed cellular GA has outperformed a standard cGA when using an array with 64 individuals. The standard cGA fails in all experimental samples while 90% hit rate was achieved by the distributed approach.

The migration policy which measures the genotypic diversity performed poorly in comparison to having maximum frequency for migration or phenotypic adaptive migration. The interaction among sub-populations is limited to individuals placed at grid corners. Yet the performance improvement is significant.

At algorithmic level, fault tolerance has been investigated in [79] and [80] and in [52] at implementation level. In particular the inherent ability of cellular GAs to efficiently retrieve the attitude parameters of a vehicle operating at aerospace environmental conditions has been assessed. In such a system the electronics are prone to faults, such as SHEs and SEUs, due to radiation. In the next subsection, fault tolerant mitigation techniques are proposed using the distributed cGA studied in this section.

3.4.4 Fault Tolerance Perspective

To extend into the fault tolerant arena the distributed cGA platform introduced in the previous section is the objective in this subsection. The algorithm encompasses speed, adaptability and performance as its key objectives while dealing with Single Hard Errors (SHE) from the fault tolerance perspective. The GPS attitude determination technique is also based on the Ambiguity Function Method (AFM) overcoming restrictions and computational overheads incurred by existing code-phase techniques.

Navigation systems need to fulfil not only time and accuracy constraints, but also fault tolerance capability must be provided. Electronic systems of air and space crafts operate at difficult environmental operation conditions, where these are prone to unexpected radiation doses. Fault tolerance has been researched following different perspectives; for example, hardened devices have been developed in order to protect electronic modules; however, low availability and consequently high costs are disadvantages of this approach. Thus, tackling fault tolerance at an algorithmic level could substitute or strengthen hardware oriented techniques. In the next subsection the fault tolerant approach proposed in this study is detailed.

Overall, the proposed approach is based on measuring the population diversity during the evolutionary process and thus to control the exploitation-exploration trade off. An adaptive control to dynamically switch from exploration to exploitation to guide the search process has been performed based on phenotypic diversity changes during the evolutionary process. Applying this adaptive approach improves the distributed cellular GA performance when Single Hard Errors (SHEs) occur during the search; the loss of diversity due to this kind of faults at fitness score registers is investigated.

Results show a superior performance for the distributed and adaptive approach with the added ability to deal with SHEs at fitness score registers. Convergence time is better maintained when applied to large array sizes. This algorithmic approach would provide an architecture for the GPS attitude determination problem which meets the goals of speed, accuracy, adaptability and performance.

3.4.4.1 Evolutionary Approach

A distributed parallel cGA with several independent sub-populations evolving with the possibility of information exchange among them during predefined time slots is implemented from a fault tolerant perspective. Having cellular structures on each sub-population has among other advantages, the fast spreading of solutions due to neighbourhoods overlapping. In Figures 2.1 and 2.2, diagrams for distributed and cellular GA approaches are drawn. An important characteristic of distributed and cellular GAs are the super-linear speedups they can achieve due to their massive parallelism, a detailed review of PGAs speedups is provided in Subsection 2.4.2. Among others, those abilities make distributed cGAs suitable to deal with the difficulties the GPS attitude determination problem presents.

In cellular GAs, the neighbourhood to grid ratio (NGR) determines the strength of the selection intensity with which the search is performed. Normally, it is considered that having an explorative search at the beginning followed by a more aggressive or exploitative search, is an appropriate way to deal with hard landscapes. However, the selection intensity in most evolutionary techniques remains constant during the search, unless genetic operation parameters are modified at run time. Yet, cellular GAs possess structural properties that can be dynamically modified and thus varying the induced selection pressure.

The same configuration topology presented in Figure 3.14 is used. A structured population implemented on a square lattice is divided into sub-populations. Each sub-population is connected through individuals located at the corners. However, the independent evolution of sub-populations is prioritized. The interaction among sub-populations is ruled by several migration policies which have been defined and empirically assessed. The fault tolerant response of the distributed cGA is based on its ability to react when phenotypic diversity is lost because of SHEs affecting its fitness score registers. Therefore, migration acts as a regulator to allow certain individuals to migrate to those sub-populations affected by faults.

In genetic algorithms the phenotypic space is where solutions are represented by their fitness values. When SHEs affect the fitness score registers triggering either faulty low or high fitness scores due to stuck at zero or one single or multiple bit flips, weak or strong individuals can be wrongly selected or discarded from the evolutionary process and negatively impact the outcome of the search.

As mentioned before, defining adequate migration policies is one of the main issues in parallel GAs. Applying the maximum migration frequency (migration is performed every generation) could lead the search to an exploration excess, where individuals would spend time on areas of the search space possibly with poor solutions. The adaptive approach measures the way in which phenotypic values are affected by SHEs and promotes the exchange of predefined individuals through migration. Algorithm 7 describes the distributed cGA using a population placed on a square grid which is subdivided in 4 sub-populations, with a L5 local neighbourhood. The overall condition for the migration of individuals is defined by the average fitness score difference in the current and previous generations. This change is defined by $\Delta \bar{f}_t$ and $\Delta \bar{f_{t-1}}$. An increase in the overall fitness score by a factor $(1 - \epsilon)$ indicates that the search process is going fast, with the risk of quick loss of diversity and stagnation; this possibility is increased by SHEs at fitness score registers. Promoting exploration through migration is a way to overcome both effects and lead the search to the global optimum. On the contrary, if this difference decreases by the same factor (ϵ), exploitation should be promoted internally in each sub-population. Low computational cost is an advantage of the adaptive approach; the complexity to calculate the average fitness of a population of n individuals is $O(n)$. In the next subsection, the experimental constraints and the results analysis are presented.

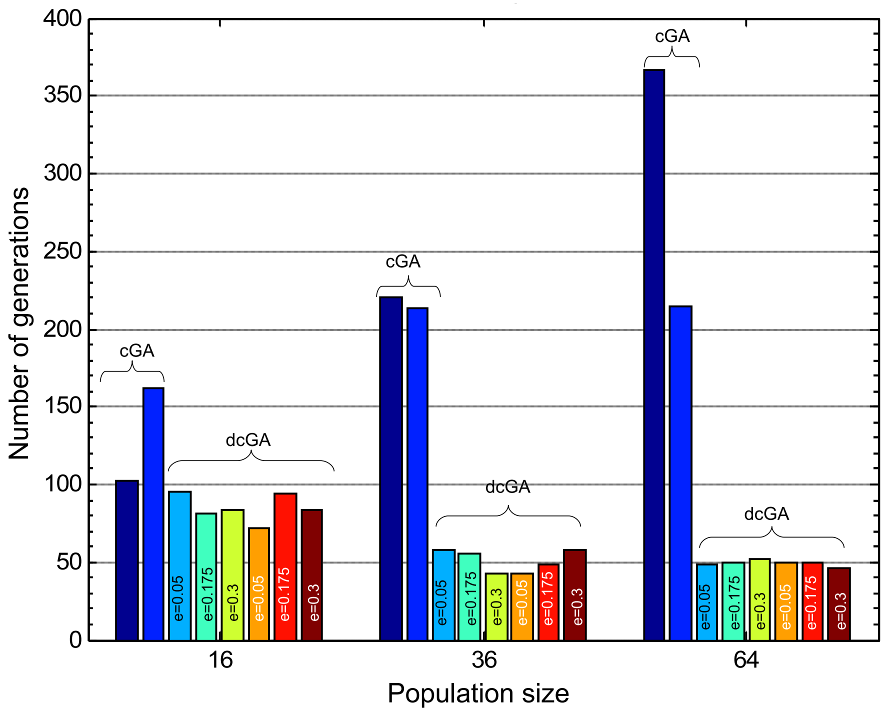
3.4.4.2 Results Analysis

The same experimental constraints defined in Subsection 3.4.2 are evaluated. For comparison, the performance of a standard cGA is used as a reference. The stop condition forces most of the individuals to have achieved a solution of the same quality. A threshold of 0.99 of the population's average fitness score is the stop condition. This threshold is empirically determined in non faulty conditions presenting a hit rate of 50%, 25% and less than 5% for 16, 36 and 64 individuals respectively.

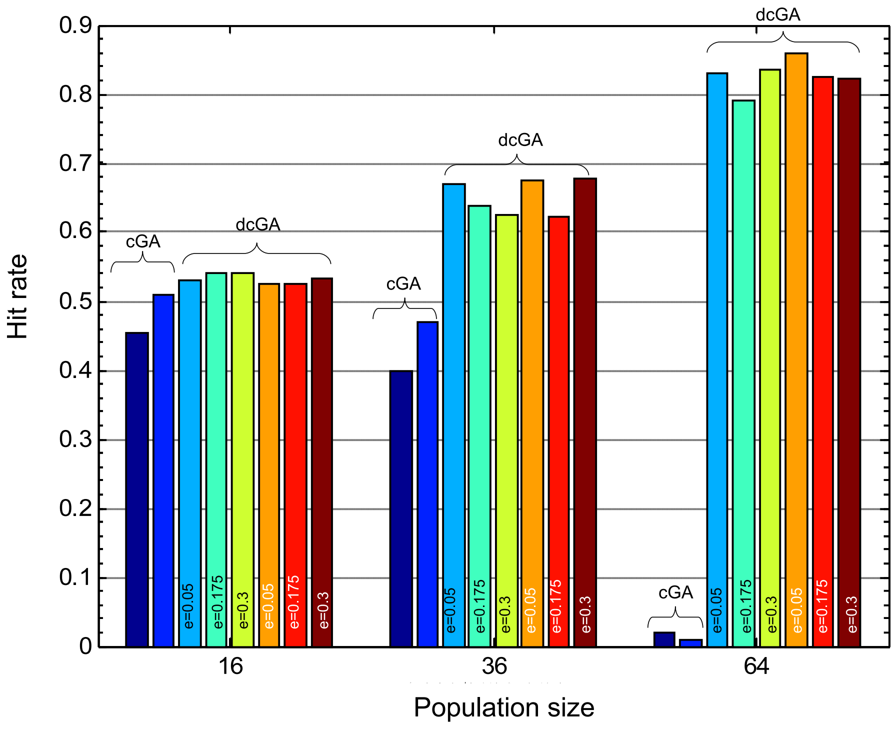
The first faulty scenario is stuck at zero SHEs at fitness score registers. The same random faults distributions are injected in all independent runs. For experimental purposes three values for the ϵ factor are evaluated. When $\epsilon = 0.05$ migration scarcely occurs. In contrast, $\epsilon = 0.3$ is a more flexible condition for individuals to migrate. In the experimental set-up, an intermediate value has also been experimentally assessed: $\epsilon = 0.175$. The ϵ factor was originally introduced by Dorronsoro et al. in [6].

A total of 200 independent runs are performed for each experimental case with up to 25% of faulty individuals. In Figure 3.17(a), the average number of generations is presented for three population sizes. In general, the distributed cGA that measures the effect of SHEs in the phenotypic space outperforms the standard cGA. In the figure, from left to right the first two blue columns correspond to the simulation of the standard cGA and the rest to the distributed cGA approach. The convergence time changes according to the population size. The condition to fullfil if migration were to happen with $\epsilon = 0.05$ is tougher. Therefore this configuration provides in general a better convergence time for all population sizes. In terms of the hit rate, the largest population with 64 individuals outperforms smaller arrays. In particular, the distributed cGA significantly improves the hit rate of the standard cGA. The improvement is a consequence of being able to sort out the scenario of possibly having strong individuals with faulty fitness scores which are not selected for mating.

In the second faulty scenario with stuck at logic one bits at fitness score registers, the same experimental constraints are evaluated. However, the amount of faulty individuals is reduced to 10%, because this case is more difficult to deal with. The selection of faulty individuals causes that possibly weak individuals are quickly spread throughout



(a) Average number of generations



(b) Hit rate

Figure 3.17: Stuck at zero faults at fitness score registers

local subpopulations and among subpopulations.

It was mentioned before that the same grid configuration and migration policies are assessed in this subsection from a fault tolerant perspective. The performance of the standard cGA is negatively affected. Low hit rates are reported, from the total number of independent runs, only 5% converge. The average number of generations to find the global optima for the proposed distributed cGA increases to 250 generations and the hit rate is 50% for a population size of 64 individuals. These results provide a reference of the criticality of this faulty scenario as the hit rate is not acceptable for the GPS attitude determination problem.

Although, the distributed cGA outperforms the standard cGA, those results only provide an overview of the criticality in this faulty scenario and the proposed algorithmic approach is a base for further investigation.

In the last faulty scenario, bits at fitness score registers are stuck at zero or one logic (hybrid case). Faulty zero and one bits are induced with the same probability. Therefore, strong and weak individuals are wrongly selected for reproduction and the offspring spread locally and then globally in the distributed cGA and only globally in the standard cGA.

The same experimental constraints are applied and results show that the standard cGA is not able to cope in this faulty scenario. Not only weak individuals are quickly spread and mated with others selected as a result of SHEs, but also possibly strong individuals are discarded for reproduction. On the other hand, the distributed cGA is able to cope with this scenario, with up to 10% of faulty chromosomes, with hit rate of 70% which is also the best hit rate, and an average processing time of less than 80 generations for the largest array. Similar to the previous case, this faulty scenario must be reassessed from a local perspective together with the benefits of adaptive migration on a distributed cGA which clearly improved a standard cGA. In the next subsection, a summary of the study is presented.

3.4.4.3 Summary

A fault tolerant and dynamic approach targeting the GPS attitude determination problem has been investigated. An evolutionary platform based on a distributed cGA has been assessed when SHEs affect the fitness score registers. The experimental results have confirmed that a distributed cGA outperforms a standard cGA in several faulty scenarios. The importance and influence of migration policies have also been investigated. The distributed cGA with an array of 64 individuals improves its convergence time and its hit rate in most faulty scenarios. For example, the hit rate is negatively affected in faulty scenarios where bits are stuck at one or a combination of stuck at zero and one bits. Although, the obtained hit rate is not adequate for safety-critical applications such as the GPS attitude determination, a distributed cGA does improve the hit rate in these critical faulty scenarios. Thus, the results obtained in this research confirm the ability of distributed cGAs to deal with SHEs in fitness scores registers.

3.5 Chapter Summary

This chapter has investigated the inherent abilities of cellular GAs to deal with SHEs and implicitly SEUs. Data critical to the system has been targeted such as fitness score and chromosome registers or memory allocations. Faulty scenarios were evaluated considering stuck at zero, stuck at one and a hybrid case with stuck at zero and one faults. The main idea is to take advantage of cGAs' structural properties to overcome faulty scenarios. The contribution to knowledge in this topic is:

- Migration operation was proposed and assessed as a mitigation technique to deal with faults affecting fitness score registers. Several migration policies were defined and evaluated based on random, best neighbourhoods and best neighbours selection.

- The configuration of the local neighbourhood was proposed and assessed as a way to provide every individual with more alternatives for selection and therefore to avoid faulty individuals. It is also used as a measure to modify the induced selective pressure for a better balance of the exploration-exploitation trade-off. Several local neighbourhoods configurations were evaluated with a minimum of 4 and a maximum of 12 neighbours.

- The proposed mitigation techniques were able to deal with randomly generated faulty scenarios within limits. Faults criticality implies that potential solutions are discarded for selection (stuck at zero) while poor solutions are selected for reproduction (stuck at one). The second case is much harder.

- Fault tolerant cGAs were able to deal with stuck at zero faults and improve algorithm's efficiency and efficacy for higher percentages of faulty individuals using large population sizes. However with stuck at one faults the hit rates in most experimental cases dropped significantly. Only through random migration the algorithm was able to cope with the faults. The hybrid faults case provided intermediate performance results.

- Another proposed and assessed mitigation technique is to measure the genotypic diversity loss due to SHEs affecting chromosomes registers and to modify the configuration of the topology as a way of providing lower selective pressure and therefore a more explorative search; and moreover to explicitly migrate individuals in order to sort out faulty individuals.

- The proposed mitigation technique was able to deal with stuck at zero SHEs at chromosomes registers while improving the algorithmic efficacy. On the other hand, with stuck at one faulty bits there was also a slight improvement in results accuracy and in the average number of generations to converge to the global optimum.

- A distributed parallel approach was proposed and assessed to improve the results accuracy. Empirical results showed an improvement of $\bar{f} = 1 \times 10^{-2}$ in the population's average fitness score that implies a variation in the attitude parameters of $\varphi = 0.05$ degrees for the azimuth angle and $\beta = 0.01$ degrees for the elevation angle.

- A fault tolerant approach was then introduced to tackle SHEs and implicitly SEUs affecting fitness score registers using the proposed distributed parallel approach. The idea behind is to balance the exploration-exploitation trade-off to deal with the loss of diversity due to the faults.

In this chapter the case study has been the GPS attitude determination problem. The operational constraints of this problem make it an interesting case for investigation from algorithmic and implementation perspectives. However, focusing on a single problem does not allow to validate the effectiveness of cGAs when tackling hard optimization problems. In the next chapter, the dynamics of cGAs are further investigated through a more complete set of test problems presenting a wide variety of characteristics and therefore different levels of difficulty.

Chapter 4

On Dynamic Cellular Genetic Algorithms

Providing a deeper insight into the flexibility cellular GAs inherently possess is the aim of this chapter. When appropriately exploited, cGAs' unique properties as regards the interaction of individuals enhance their searching abilities. In EAs, the main interest is to preserve population diversity through satisfactorily balancing the exploration and exploitation of solutions throughout the search space; in order to successfully converge to the global optimum in the minimum number of generations [81]. In this regard, structural properties of cGAs have shown their effectiveness in maintaining this balance, in several cases improving cGAs performance [31]. In this chapter, a dynamic cellular approach that *internally* reconfigures a traditional square topology into smaller sub-cellular structures is investigated.

In order to validate the importance of decentralized GAs, authors have compared their results to those of standard GAs, and to those from improved versions of standard GAs, such as the steady state GA or the generational GA. In several cases, cGAs were proved to perform better in terms of algorithmic efficiency and efficacy [41]. In subsection 2.4.4 a comparison between steady state, generational GAs and cellular GAs has been included. The effect of synchronism in migration policies of distributed approaches using cGAs or panmictic independent populations, has also been studied and compared. Asynchronous and low frequency migration in distributed PGAs outperformed synchronous migration policies [42]. On the other hand, the ease of implementation of processor elements in cGAs is also an advantage that has been investigated and implemented [53].

Alba et al. have widely studied decentralized GAs [82]. An important area covered by their work is to compare distributed parallel GAs considering homogeneous or heterogeneous populations or a mixture of these. In [21, 12] a detailed analysis of several approaches at algorithmic and implementation levels is presented. Configurations of homogeneous sub-populations (only panmictic or cellular), heterogeneous sub-populations (a mixture of panmictic and cellular), comprising several migration and replacement criteria are evaluated. In general, those results showed an improved performance of distributed cellular GAs in terms of efficacy over panmictic distributed PGAs. However, cellular GAs showed slower convergence times than standard steady-state distributed GAs. Communication among sub-populations was also assessed. Asynchronous communication leads to faster convergence and better speed-ups than synchronous ones.

As discussed in previous chapters, in cellular GAs the population is distributed in a torus like grid structure with wraparound edges. One individual is placed per grid position and can only interact with nearby neighbours. Several parameters need to be configured in order to achieve an optimum algorithmic performance, such as, the shape and size of the local neighbourhood and the population topology, the migration rate and frequency (in case explicit migration occurs), local selection and replacement policies, among others [13].

Previously, researchers have proposed to manipulate the structure of cGAs to control the selection pressure. In [83] disturbances are induced in order to create inner islands and thus modify the selective pressure. Results improved the accuracy of results but a statistical analysis was not provided. The stop condition compares the solutions quality after a certain number of generations. However, it is believed more appropriate to evaluate the algorithmic performance once a solution of the same quality has been achieved. In [6], the dynamic reconfiguration of the population topology is introduced. The proposed approach implies an explicit form of migration which occurs during the relocation of individuals. The reconfiguration of the grid is performed after a specific number of generations or adaptively. Once the lattice shape has changed, individuals interact with others previously located at non-adjacent positions and therefore diversity is promoted. In this chapter, the proposed dynamic reconfiguration mechanisms maintain the original adjacency of individuals while changing internally the configuration of the topology. Therefore any improvement in the algorithmic performance is only a consequence of the lattice reconfiguration mechanism with no intervention of other genetic operations. The proposed mechanisms are applied to a set of continuous, real and combinatorial problems presenting characteristics such as multi-modality, epistasis, deceptiveness and non-regularity.

In the next two sections, dynamic cellular GAs are investigated from a structural perspective. In Section 4.1 the structural properties of cGAs are dynamically modified in order to improve their performance. In Section 4.2 a local selection method that takes advantage of the structural characteristics of cGAs topology is studied and dynamically modified. Most benchmark problems included in Appendix A are assessed in the following sections.

4.1 Dynamic cGAs Based on Structural Properties

Cellular Genetic Algorithms (cGAs) have attracted the attention of researchers due to their high performance, ease of implementation and massive parallelism. Research has been carried out not only from an empirical point of view but also in an effort to establish a theoretical base to provide modelling tools that contribute to cGAs understanding. Maintaining an adequate balance between exploitative and explorative search is essential when studying evolutionary optimization techniques. In this respect, cGAs inherently possess a number of structural configuration parameters that are able to contribute in sustaining diversity during evolution. In this chapter, the internal reconfiguration of the lattice is proposed, to constantly or adaptively control the exploration-exploitation trade-off. Genetic operators are characterized in their simplest form since algorithmic performance is assessed based on the implemented reconfiguration techniques. Moreover, *internal* reconfiguration allows the adjacency of individuals to be maintained. Hence, any improvement in performance is only a consequence of topological changes. Two local selection methods presenting opposite selection pressures are implemented in order to evaluate the influence of the proposed techniques.

To provide a deeper insight into the great flexibility cellular GAs inherently possess, their unique properties as regards the interaction of individuals are investigated with the aim of enhancing their searching abilities. In EAs, the main interest is to preserve population diversity through satisfactorily balancing the exploration and exploitation of solutions throughout the search space; in order to successfully converge to the global optimum in the minimum number of generations [81]. In this regard, structural properties of cGAs have shown their effectiveness in maintaining this balance, in several cases improving standard cGAs performance [31]. In this study, a dynamic cellular approach that *internally* reconfigures a traditional square topology into smaller sub-cellular structures is investigated.

As a result of the population topology and the local neighbourhood configuration, the selection pressure characterized by the NG ratio, can be constantly or adaptively controlled in order to improve the performance of cellular GAs. Previously, researchers have proposed dynamically reconfiguring the population topology [6]. However, a form of migration occurs during the relocation of individuals; this approach from now on is named as *external* lattice reconfiguration. Three main scenarios were evaluated: static (fixed topology shape), pre-programmed (reconfiguration at predefined time slots) and adaptive (dynamic reconfiguration). Statistically significant results were obtained for several difficult benchmark problems. The mapping of individuals, after grid rearrangement, induces a loss of the natural adjacency of individuals. In this chapter, adjacency among individuals is maintained while constant or adaptive lattice reconfiguration is carried out by subdividing the entire population into smaller square, rectangular or linear toroidal arrays, with no induced migration among them apart from the neighbourhoods overlapping. The proposed *internal* reconfiguration mechanisms are applied to a set of continuous, real and combinatorial problems covering characteristics such as multi-modality, epistasis, deceptiveness and non-regularity.

4.1.1 Selection Pressure Control

The selection pressure is controlled constantly or adaptively through the internal reconfiguration of the population topology. *Internal* reconfiguration means that there is no alteration to the natural adjacency of individuals. In cellular GAs, the number of generations the best individual of an initial population needs to spread its solution throughout the grid under selection operation only, indicates the intensity of the selective pressure [31, 32].

Selection pressure in cGAs provides important information about the evolutionary process showing different levels of exploration or exploitation with which population topologies contribute to the search. In [82] selection pressure is considered as an indicator of the search speed. If the average fitness increases drastically from one generation

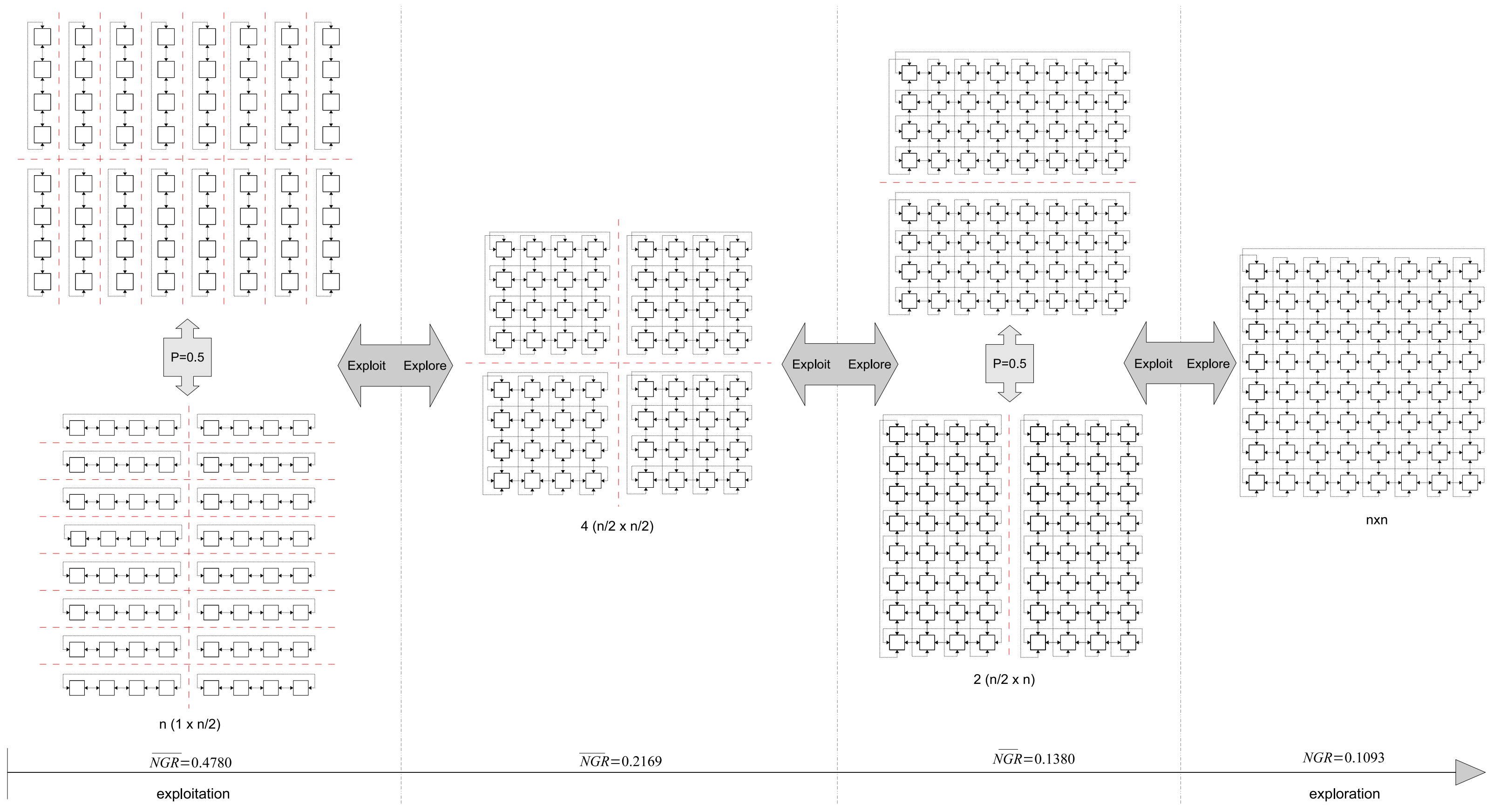


Figure 4.1: The average neighbourhood-grid ratio changes according to the lattice configuration. Performing a more explorative (lower average ratios) or more exploitative (higher average ratios) search. Rectagular and linear topologies are executed in horizontal and vertical alignments with 50% probability each.

to the other, the search process proceeds rapidly and therefore exploitation is high. On the other hand, if the average fitness of the population is the same or has decreased compared to previous generations, it means the population is slowly evolving. Thus individuals are widely spread throughout far-off regions of the search space. Although, this analysis covers several possible scenarios, it does not include all of them. For example, individuals can be loosely located over the landscape and their average fitness could be increasing or vice versa.

Researchers have demonstrated that the exploration-exploitation trade-off can be implicitly controlled in cGAs through the configuration of the population topology and the local neighbourhood [46, 26]. A measure known as the neighbourhood-grid ratio (NGR) has been defined in order to model this behaviour. In Figure 4.1, four population configurations are drawn. Internally, smaller cellular structures imply a higher or lower NGR, that as a whole would modify the search making it more exploitative or explorative. Configurations denoted by $n\left(1 \times \frac{n}{2}\right)$ and $2\left(\frac{n}{2} \times n\right)$ in Figure 4.1, are considered in both vertical and horizontal alignments. These lattice configurations have been used in the experimental set-up.

The NGR was first introduced by De Jong in [31]. Details of how the NGR is calculated are provided in Chapter 2, subsection 2.4.1. For most topologies in Figure 4.1, a compact Von Neumann local neighbourhood is used consisting of individuals located at North, East, South and West with Manhattan distance of one from the central individual; except in linear lattices ($n\left(1 \times \frac{n}{2}\right)$) that employ a linear local neighbourhood with Manhattan distance of two.

For a population of 400 individuals, the dispersion measures for the Von Neumman and the linear local neighbourhoods are $D_{vonNeumman} = 0.8944$ and $D_{linear} = 1.4142$ respectively. The corresponding average NGR for the cellular configurations drawn in Figure 4.1 are as follows:

- $NGR_{n \times n} = 0.1093$
- $N\bar{G}R_{2\left(\frac{n}{2} \times n\right)} = 0.1380$
- $N\bar{G}R_{4\left(\frac{n}{2} \times \frac{n}{2}\right)} = 0.2169$
- $N\bar{G}R_{n\left(1 \times \frac{n}{2}\right)} = 0.4780$

A square topology $(n \times n)$ is more explorative (lower ratio), while dividing the population in two rectangular grids $(2\left(\frac{n}{2}\times n\right))$ results in having more exploitative sub-arrays with in average higher NGR.

Having different ratios means having higher or lower selection pressures. Therefore, decentralized GAs are structurally capable of modifying the selection pressure while changing the population structural configuration. Previously, Dorronsoro et al. proposed several constant and adaptive criteria to *externally* modify the induced selection pressure in order to improve the performance of cGAs [6]. However, that approach implies an inherent migration mechanism; through it, new positions are calculated for individuals when the population topology is changed among square, rectangular and narrow lattice shapes. Although results showed an improvement in performance for several benchmark problems, it is not clear if this improvement is purely due to controlling the NG ratio or if it is also a consequence of migration. Controlling, constantly or adaptively, the NG ratio while maintaining the original adjacency among individuals is the aim in this study.

In order to characterize selection pressure in cellular GAs, the take-over time concept is used. This is also referred, in cGAs, as the proportional growth rate of the best individual. The take-over time reflects how long it takes for the best individual to spread its solution throughout the whole lattice, applying only local selection. Thus, longer take-over times represent lower selection pressure and therefore more explorative behaviour. In contrast, shorter take-over times correspond to higher selection pressure, equivalent to a more exploitative search. More details on take-over time theory are included in Chapter 2, Subsection 2.4.1.

At a local level, two different selection methods are applied; widely known binary tournament selection and ad-hoc for cellular GAs, anisotropic selection proposed by Simoncini et al. [84, 29]. These selection methods provide distinctively opposite selective pressures. Binary tournament is highly exploitative in comparison to anisotropic selection which is more explorative.

In Figure 4.2, the proportional growth curves for both selection methods are shown. One hundred experiments for a population of 400 individuals were performed. Only constant lattice reconfiguration among topologies drawn in Figure 4.1 is performed. Constant reconfiguration occurs every 5 generations. Once an initial population is created, evolution starts on a $4\left(\frac{n}{2}\times\frac{n}{2}\right)$ topology. Thereafter, every 5 generations a different lattice reconfiguration performs the search, in a cycle from high/middle to lower selection pressure. For $n\left(1 \times \frac{n}{2}\right)$ and $2\left(\frac{n}{2} \times n\right)$ formations, horizontal or vertical alignments are executed with probability $P = 0.5$. Adaptive lattice reconfiguration is not subject to proportional growth analysis since it depends on the diversity of phenotypes or genotypes; information that is not available if only local selection is applied.

In a $n \times n$ square topology, binary tournament induces higher selection pressure than anisotropic selection which is more explorative. In Figure 4.2, corresponding take-over time curves for binary tournament are much closer than anisotropic selection curves. Therefore, anisotropic selection through constant lattice reconfiguration allows a wider span to dynamically adjust the selective pressure in comparison to binary tournament selection. Moreover, on a static square topology, anisotropic selection is more explorative when compared to constantly reconfiguring the grid. This behaviour is not repeated through binary tournament selection, presenting more explorative behaviour when constant reconfiguration is performed. In the following section, the details of the algorithmic procedure and the diversity measures for lattice reconfiguration are provided.

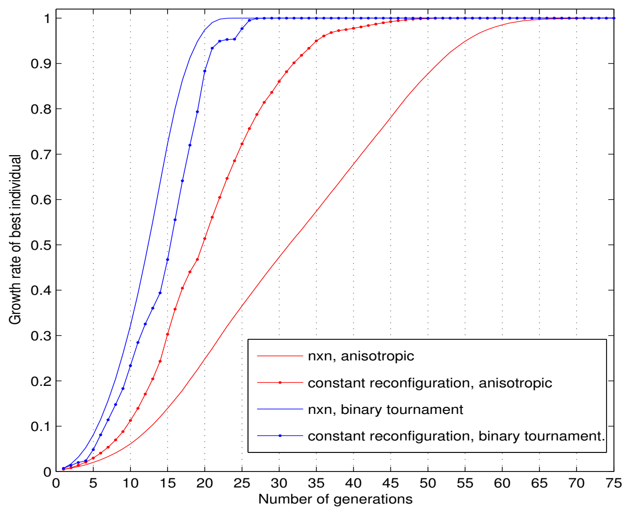


Figure 4.2: Take-over time for binary tournament and anisotropic local selection methods. Constant reconfiguration is performed using the four topologies drawn in Figure 4.1

4.1.2 Cellular Configuration

This study proposes to constantly or adaptively reconfigure the internal population topology to inherently tune the neighbourhood to grid ratio (NGR), and thus achieve an appropriate balance between searching exploration and exploitation, essential for maintaining population diversity during evolution. In this proposal, lattice reconfiguration is implemented by subdividing a squared population into smaller squared, rectangular or linear toroidal arrays. A similar approach has been referred to as a parallel cellular GA [82]. However, that approach has a fixed *internal* configuration. Whilst an homogeneous distributed or an island cGA model is one possible perspective [85], migration policies have not been investigated nor implemented. During constant or adaptive periods of evolution, smaller topologies maintain internally toroidal connections [18, 86].

In Algorithm 8, the pseudocode for the proposed cellular GA approach is included. A single random seed is used to generate the entire initial population. Two topology configurations are evaluated to start the search process; the population is initially evolved on a $4\left(\frac{2}{n} \times \frac{2}{n}\right)$ or a $n \times n$ topology configuration (line 2). For each individual, neighbours located at North, East, South and West positions are evaluated in order to select a second parent for reproduction (line 7). Two local selection methods presenting opposite selection pressure are applied for experimental purposes. Binary tournament presents higher selection pressure than anisotropic selection. After synchronous updating (line 16), the population's average fitness score is verified as the stop condition (line 17), if the problem specific threshold has not been achieved, lattice reconfiguration is performed either constantly or adaptively based on phenotypic or genotypic diversity measures (line 22).

Binary tournament randomly chooses two individuals from the local neighbourhood and the best one (x') is mated with the central individual (x_0). On the other hand, anisotropic selection requires an extra parameter (α) that leads the search in North-South or East-West grid directions. Thus, anisotropic probabilistic equations are defined for each individual in the neighbourhood as: $P_N = P_S = P_0\,(1+\alpha)$ and $P_E = P_W = P_0\,(1-\alpha)$; where P_0 is the uniform probability for each neighbour to be selected. If $\alpha = 0.8$ and $P_0 = 0.25$, individuals at North and South positions are assigned with higher probabilities for selection compared to East and West individuals that remain with lower selection probabilities. In this way the search process is directionally

Algorithm 8 Reconfigurable cGA

```
1: procedure cGA
2:     $n \times n \ | \ 4(\frac{n}{2} \times \frac{n}{2})$                ▷ Initial topology configuration
3:     for $k \leftarrow 1, gens$ do
4:         for $i \leftarrow 1, m$ do
5:             for $j \leftarrow 1, n$ do
6:                 $x_0 = x(i,j); f_0 = f(i,j);$
7:                 $(f_n, f_e, f_s, f_w) \leftarrow evaluation(x_n, x_e, x_s, x_w);$
8:                 $(x') \leftarrow selection(x_n, x_e, x_s, x_w);$
9:                 $(x_1, x_2) \leftarrow recombination(x_0, x');$
10:                $(x'_1, x'_2) \leftarrow mutation(x_1, x_2);$
11:                $(f'_1, f'_2) \leftarrow evaluation(x'_1, x'_2);$
12:                $(f'_0) \leftarrow [max|min](f_0, f'_1, f'_2);$
13:                $(x_{temp}(i,j), f_{temp}(i,j)) \leftarrow replace(x'_0, f'_0);$   ▷ Replace policy *if-better*
14:            end for
15:        end for
16:        $x = x_{temp}, f = f_{temp};$                ▷ Synchronous updating
17:        if $\bar{f} <= threshold$ then
18:            stop;
19:        else
20:            next;
21:        end if
22:        $CLR \mid PLR(f) \mid GLR(x);$        ▷ Execute lattice reconfiguration mechanism
23:    end for
24: end procedure
```

guided and adjusting the α parameter would supply higher or lower selection pressure [87, 84]. For experimental purposes, $\alpha = 0.8$ has been used due to presenting lower selection pressure and thus being more explorative. Although fewer studies have been reported using anisotropic selection in cGAs, this method does provide more flexibility in terms of selection pressure while taking advantage of cGAs structural properties with the added effect of the reconfiguration mechanisms proposed in this research.

Recombination is performed using Single Point Crossover (SPC) with the highest probability and constant low mutation probability. Both genetic operators have been applied in their simplest form, as it is not the objective of this research to evaluate the effect of either of them but to widely investigate the effect of dynamically changing selection pressure through cGAs structural properties. Although it is outwith the scope of this research, particular attention should be paid to mutation as its effect has been modelled for cellular GAs as an essential control parameter to lead or mislead the search [76, 77].

4.1.2.1 Constant Lattice Reconfiguration

In Algorithm 8, line 22, $[CLR|PLR(f)|GLR(x)]$ corresponds to Constant Lattice Reconfiguration, Phenotypic Lattice Reconfiguration or Genotypic Lattice Reconfiguration, respectively.

CLR is detailed in Algorithm 9. A constant interval of c number of generations is defined for lattice reconfiguration to be performed (line 2). In both experimental set-ups, c is set to 5 generations. That interval was selected following the experimental constraints implemented in [32] where GP island models were evaluated with larger population sizes. However, the constant frequency for cellular sub-structures to reconfigure the overall population topology is a free parameter that can be empirically assessed. For rectangular and linear grid configurations (see Figure 4.1), both horizontal and vertical alignments are performed each with 50% probability. Thus the NGR is constantly decreased or increased in a cyclical pattern.

4.1.2.2 Phenotypic Lattice Reconfiguration

PLR procedure to evaluate phenotypic diversity is shown in Algorithm 10. Initially, to compare the proposed *internal* reconfiguration approach, the same phenotypic measure defined in [6] is evaluated in the first experimental set-up, see Subsection 4.1.3.

Algorithm 9 Constant lattice reconfiguration

```
1: procedure CLR
2:     if $k \ mod \ c \ == \ 0$ then                                   ▷ c is a constant
3:         if $NGR = NGR_{n \times n}$ then                              ▷ Lowest NGR
4:             $NGR = NGR_{n\left(1 \times \frac{n}{2}\right)}$;        ▷ Reconfigure the grid to the highest NGR
5:         else
6:             $NGR = \left[NGR_{2\left(\frac{n}{2} \times n\right)} | NGR_{4\left(\frac{n}{2} \times \frac{n}{2}\right)}\right]$;   ▷ Middle NGR
7:         end if
8:     else
9:     end if
10: end procedure
```

Dorronsoro et al. considered the average fitness score as a way to evaluate the phenotypic diversity; yet, an extra tuning parameter (ϵ) was employed to measure the loss of diversity. A value of $\epsilon = 0.05$ was empirically determined in [6] as an optimum. It is considered that too much exploitation would represent large changes on phenotypic values, that is most individuals are closely distributed over the landscape. On the contrary, excessive exploration would lead individuals to distant and possibly poor regions of the search space. Both states are not desirable, instead a balance between exploration and exploitation should be achieved. As explained before, this criterion does not cope with all possible scenarios. For example, individuals could be located at distant regions of the landscape and yet present significant changes in their fitness scores. This example is evident in multi-modal search spaces where the evolutionary process could stagnate at local optima. To support this idea, in [88] a study on diversity in the representation space is presented. Phenotypic diversity is measured as the number of unique fitness values in the population at a certain time known as the phenotypic entropy. Consequently, in the second experimental set-up in Section 4.1.4, the phenotypic entropy is calculated as a condition for the reconfiguration of the lattice. Both strategies for measuring phenotypic diversity are defined next:

- The average population fitness score, $\Delta\bar{f}$, is measured in consecutive generations. Thus, the difference among $\Delta\bar{f}_t$, $\Delta\bar{f}_{t-1}$ and $\Delta\bar{f}_{t-2}$ would determine if the phenotypic diversity has or has not significantly changed when comparing the fitness of current ($\bar{f}$)and previous generations ($\bar{f}_{t-1}$,$\bar{f}_{t-2}$); where $\Delta\bar{f}_t = \bar{f}_t - \bar{f}_{t-1}$.

If $\left(\Delta\bar{f}_t - \Delta\bar{f}_{t-1}\right)$ is less than $\left(\Delta\bar{f}_{t-1} - \Delta\bar{f}_{t-2}\right)$, individuals are spread and exploitation should be promoted by increasing the average NGR. If the same condition indicates an increase in the average fitness score, exploration should be raised through switching to a lattice configuration that presents an average lower NGR. Otherwise, the population topology remains the same.

- The phenotypic entropy, H_P, among consecutive generations:

$$H_P = -\sum_{j=1}^{N} f_j \log\left(f_j\right) \tag{4.1}$$

where f_j is the proportion of individuals in one generation having fitness j. Thus, the difference between ΔH_{P_t}, $\Delta H_{P_{t-1}}$ and $\Delta H_{P_{t-2}}$ would determine if the phenotypic diversity in the current generation has or has not significantly changed with respect to previous generations; where $\Delta H_{P_t} = H_{P_t} - H_{P_{t-1}}$.

If $\left(\Delta H_{P_t} - \Delta H_{P_{t-1}}\right)$ is less than $\left(\Delta H_{P_{t-1}} - \Delta H_{P_{t-2}}\right)$, diversity of phenotypes has decreased, and exploration should be encouraged through switching to a lattice configuration that presents on average a lower NG ratio. If the same condition indicates an increase in phenotypic diversity, in order to make the search more aggressive, exploitation should be promoted by increasing the average NG ratio. Otherwise, the population topology would remain the same.

4.1.2.3 Genotypic Lattice Reconfiguration

In order to determine genotypic diversity, the Hamming distance among chromosomes is used as a measure. In Algorithm 11, details for GLR procedure are presented. In line 2, the distance between current ($\Delta\bar{H}_{G_t} = \bar{H}_{G_t} - \bar{H}_{G_{t-1}}$) and previous ($\Delta\bar{H}_{G_{t-1}}$, $\Delta\bar{H}_{G_{t-2}}$) generations is calculated, in order to assess genotypic diversity changes.

In a similar way to phenotypic diversity, if $\left(\Delta\bar{H}_{G_t} - \Delta\bar{H}_{G_{t-1}}\right)$ difference is less than $\left(\Delta\bar{H}_{G_{t-1}} - \Delta\bar{H}_{G_{t-2}}\right)$, exploration should be encouraged through a lattice configuration with an average lower ratio. On contrast, exploitation should be promoted through increasing the NG ratio and having more exploitative cellular structures. If neither of these conditions are satisfied, the population topology remains the same.

In [6], conditions for lattice *external* reconfiguration imply the relocation of individuals to new positions, inducing a kind of migration among them. Hence, the impact

Algorithm 10 Phenotypic lattice reconfiguration based on the phenotypic entropy ΔH_p or the average fitness score $\Delta \bar{f}$ measurements

```
 1: procedure PLR($f$)
 2:     if $[\Delta H_{P_t} < (2 * \Delta H_{P_{t-1}}) - \Delta H_{P_{t-2}} \mid \Delta \bar{f}_t > (2 * \Delta \bar{f}_{t-1}) - \Delta \bar{f}_{t-2}]$ then
 3:         //Promote exploration
 4:         if $NGR = NGR_{n(1 \times \frac{n}{2})}$ then                    ▷ Highest $NGR$
 5:             $NGR = NGR_{4(\frac{n}{2} \times \frac{n}{2})}$;
 6:         else if $NGR = NGR_{4(\frac{n}{2} \times \frac{n}{2})}$ then
 7:             $NGR = NGR_{2(\frac{n}{2} \times n)}$;
 8:         else
 9:             $NGR = NGR_{(n \times n)}$;
10:         end if
11:     else if $[\Delta H_{P_t} > (2 * \Delta H_{P_{t-1}}) - \Delta H_{P_{t-2}} \mid \Delta \bar{f}_t < (2 * \Delta \bar{f}_{t-1}) - \Delta \bar{f}_{t-2}]$ then
12:         //Promote exploitation
13:         if $NGR = NGR_{(n \times n)}$ then                    ▷ Lowest $NGR$
14:             $NGR = NGR_{2(\frac{n}{2} \times n)}$;
15:         else if $NGR = NGR_{2(\frac{n}{2} \times n)}$ then
16:             $NGR = NGR_{4(\frac{n}{2} \times \frac{n}{2})}$;
17:         else
18:             $NGR = NGR_{n(1 \times \frac{n}{2})}$;
19:         end if
20:     else
21:     end if
22: end procedure
```

Algorithm 11 Genotypic lattice reconfiguration

```
1: procedure GLR($x$)
2:     if $\Delta \bar{H_{G_t}} < (2 * \Delta \bar{H_{G_{t-1}}}) - \Delta \bar{H_{G_{t-2}}}$ then
3:         //Promote exploration
4:         if $NGR = NGR_{n\left(1 \times \frac{n}{2}\right)}$ then                ▷ Highest $NGR$
5:             $NGR = NGR_{4\left(\frac{n}{2} \times \frac{n}{2}\right)}$;
6:         else if $NGR = NGR_{4\left(\frac{n}{2} \times \frac{n}{2}\right)}$ then
7:             $NGR = NGR_{2\left(\frac{n}{2} \times n\right)}$;
8:         else
9:             $NGR = NGR_{(n \times n)}$;
10:        end if
11:    else if $\Delta \bar{H_{G_t}} > (2 * \Delta \bar{H_{G_{t-1}}}) - \Delta \bar{H_{G_{t-2}}}$ then
12:        //Promote exploitation
13:        if $NGR = NGR_{(n \times n)}$ then                ▷ Lowest $NGR$
14:            $NGR = NGR_{2\left(\frac{n}{2} \times n\right)}$;
15:        else if $NGR = NGR_{2\left(\frac{n}{2} \times n\right)}$ then
16:            $NGR = NGR_{4\left(\frac{n}{2} \times \frac{n}{2}\right)}$;
17:        else
18:            $NGR = NGR_{n\left(1 \times \frac{n}{2}\right)}$;
19:        end if
20:    else
21:    end if
22: end procedure
```

of reconfiguring the grid is not completely clear; as migration is an influential operation that can by itself change significantly the performance of cGAs [20, 41]. The reconfiguration mechanisms proposed here, do not induce any explicit migration and once *internally* the population topology is reconfigured, the toroidal connection with wraparound edges is maintained in all cellular sub-structures. Thus, different selection pressures are locally applied while maintaining individual adjacency during evolution. Moreover, the stop condition in every generation, evaluates the average fitness score for the entire population; in consequence the calculated diversity, either phenotypic or genotypic, relates to all individuals as a whole.

The pre-programmed criteria of Dorronsoro et al. consists of *externally* changing the topology shape after a certain number of generations, that is half way through a normal successful run for a square topology. For the adaptive approach, the authors determined an $\epsilon = 0.05$ parameter as an optimum value to measure phenotypic and genotypic diversity to switch among square, rectangular and narrow topologies. The criteria proposed here avoid the use of an extra parameter and directly measure significant changes while extending the generation-range to measure diversity. Combining diversity measures in both phenotype and genotype spaces can strengthen conditions for lattice reconfiguration as demonstrated in [6]. In the following subsection, the first experimental set-up is presented.

4.1.3 First Experimental Set-Up: From Single-Static to Several-Dynamic Lattice Topologies

The first experimental set-up aims to evaluate the effect of gradually introducing different population topologies that in average will induce different levels of exploitation, and carry out a comparison with the performance of a standard cGA on a static and fully connected topology. In Figure 4.1, each topology configuration case is drawn. Dynamically swapping between two, three and four of these configurations is empirically evaluated in order to determine if there is an improvement in the performance of the dynamic cGAs. The following cases are defined for experimental purposes:

1. Double configuration: $4(\frac{n}{2} \times \frac{n}{2}) \leftrightarrow n \times n$,
2. Triple configuration: $2(\frac{n}{2} \times n) \leftrightarrow 4(\frac{n}{2} \times \frac{n}{2}) \leftrightarrow n \times n$,
3. Quadruple configuration: $n(1 \times n) \leftrightarrow 2(\frac{n}{2} \times n) \leftrightarrow 4(\frac{n}{2} \times \frac{n}{2}) \leftrightarrow n \times n$.

Topology configurations $n(1 \times n)$ and $2(\frac{n}{2} \times n)$ are executed in horizontal and vertical alignment. The dynamic reconfiguration of the population topology is performed constantly, every certain number of generations, or adaptively based on phenotypic and genotypic diversity changes.

To support the results, the following statistical analysis is carried out, an initial normality test is performed on each set of experimental results regarding the convergence time. The Lilliefors test is used at 5% of significance. This test is suitable when a fully-specified null distribution is unknown, contrary to the Kolmorov-Smirnov test. Once the normality of results has been established, the Analysis of Variance (ANOVA) is applied to the results which follow a normal distribution whereas the Kruskal-Wallis test is applied to the results which do not follow a normal distribution.

The statistical significance is evaluated in such a way that a standard cGA with a static square topology is compared to dynamic cellular GAs with constant or adaptive lattice reconfiguration mechanisms. Therefore, a population size n distributed on a square topology formed by $\sqrt{n} \times \sqrt{n}$ individuals is compared to the three topology cases described before.

In the results tables, + symbol represents that 5% statistical significant difference is proved among convergence time results. In contrast, a • symbol indicates that convergence time results are not statistically different, and therefore applying the proposed dynamic criteria makes no difference in terms of the number of generations. In the analysis of results each test problem is individually discussed. The statistical analysis only assesses the efficiency of proposed techniques.

The following experimental constraints are evaluated:

- A population size of 400 individuals was used for most problems. Due to their size, GPS and MTTP problems are tackled using a population size of 64 and 100 individuals respectively.

- Local neighbourhood configuration is Von Neumann or linear composed by four individuals plus a central individual.

- One hundred independent runs are carried out per experimental reconfiguration criterion.

- A problem specific threshold based on the population's average fitness is used as a stop condition.

- A limit of 500 generations is used in most problems, with the exception of the Langerman function, the SLE and the MMDP problems with a limit of 700 generations.

- The diversity measures in the adaptive approaches are calculated for the entire population regardless the *internal* topology configuration.

4.1.3.1 Results Analysis

For presentation, results are divided by local selection method and problems are organized in two sets, a) continuous problems and b) real-world and combinatorial problems.

Continuous Problems

Table 4.1 presents results for binary tournament selection. The hit rate is the number of experiments that successfully converge to the threshold and for the Rastrigin function is 100% for all configurations, and the convergence time is reduced when lattice reconfiguration is applied. For all proposed reconfiguration cases, the average number of generations decreases, although this difference is more significant by the quadruple configuration alternative. Moreover, in all experimental sets statistical difference is proved. The Rastrigin function is highly multimodal, separable and regular, although difficult it does not present neither epistatic nor asymmetric characteristics.

In the Griewank function, the effect of varying the NGR through the topology configuration slightly benefits the convergence time (with statistical proof only for quadruple case) but negatively affects the hit rate which is reduced from 90% to 67% in the worst case scenario, having a double topology configuration and adaptive genotypic lattice reconfiguration. This function is highly epistatic apart from being multi-modal. Having more than one topology option combined with the high selection pressure of binary tournament negatively affects the algorithmic efficacy.

The Langerman function is more difficult in comparison to the previous problems. It is not regular which means the local optima are randomly distributed; therefore there is no advantage of having the global optima located at the same variables values. Although experiments were carried out for a problem size of 10 dimensions, the algorithm efficacy, in terms of hit rate, dropped dramatically. This does not allow to properly evaluate the proposed test scenarios. Therefore, the results of Table 4.1 correspond to 5 dimensions. Similar to previous functions, the average convergence time is improved

Table 4.1: Convergence time[1] and hit rate[2] results for continuous problems. Local selection: central + binary tournament.

TC | RM [3]	Rastrigin	Griewank	Langerman
Square | None	136.69 ± 12.42	275.65 ± 44.56	76.90 ± 20.81
	100%	**90%**	81%
Double | Constant	130.69 ± 13.44	278.80 ± 55.20	69.93 ± 12.78
	100%	85%	81%
Double | Phenotypic	127.62 ± 14.65	269.65 ± 63.96	70.12 ± 13.95
	100%	79%	86%
Double | Genotypic	127.58 ± 15.14	255.48 ± 40.10	70.01 ± 17.03
	100%	67%	**94%**
ANOVA/K-W	(+)	●	(+)
Triple | Constant	130.89 ± 14.41	263.28 ± 46.60	73.67 ± 33.92
	100%	81%	86%
Triple | Phenotypic	129.74 ± 11.73	269.60 ± 56.54	69.30 ± 15.67
	100%	79%	88%
Triple | Genotypic	130.59 ± 18.44	268.23 ± 59.83	69.69 ± 18.17
	100%	75%	88%
ANOVA/K-W	(+)	●	(+)
Quadruple | Constant	$\mathbf{124.08} \pm 13.16$	$\mathbf{257.23} \pm 57.03$	65.38 ± 13.70
	100%	68%	88%
Quadruple | Phenotypic	129.13 ± 14.62	270.54 ± 55.15	$\mathbf{65.21} \pm 15.20$
	100%	77%	85%
Quadruple | Genotypic	126.35 ± 13.56	260.72 ± 45.38	72.86 ± 26.33
	100%	77%	88%
ANOVA/K-W	(+)	(+)	(+)

1 Convergence time is measured as the average number of generations for successful experiments. Corresponding standard deviation is included after ± symbol.

2 Hit rate is presented as the percentage of successful experiments out of the total number of experiments which is one hundred samples.

3 Topology Configuration (TC) refers to the number of configurations among which lattice reconfiguration is performed, see Subsection 4.1.3. Reconfiguration Mode (RM) refers to constant or adaptive lattice reconfiguration mechanisms.

through the proposed reconfiguration mechanisms; this is more noticeable when the quadruple topology configuration is applied. The hit rate is also positively affected improving from 81% to 94% when the double topology option is performed based on genotypic diversity. The statistical analysis shows a significant difference in all test scenarios.

Table 4.2 presents the performance results for the same functions but through anisotropic selection. For the Rastrigin function, similar to binary tournament selection, the algorithmic efficiency in terms of convergence time is improved when reconfiguration is performed. There is statistical significance in all scenarios while the hit rate is maintained in 100%. The standard deviation shows results consistency among experimental sets.

Anisotropic selection presents lower selection pressure and more flexibility while having more than one topology option as explained in Section 4.1.1. The benefit of applying higher or lower selection pressure during the evolutionary process is evident for the Griewank function in both performance measures. In terms of efficacy, the hit rate improves from 67% for a cellular GA with static square topology (constant selection pressure) to 93% through constant lattice reconfiguration and having four topology configurations. Moreover, the average number of generations is reduced in 10% while statistical significance is shown in all experimental groups. On the other hand, the Langerman function presents a slight improvement in terms of hit rate (3%) while the average convergence time is reduced by approximately 30% when compared to a cGA with a static square topology.

The effect of having dynamic selection pressure is noticeable for both local selection methods. Convergence time is improved through binary local selection, although the hit rate is negatively affected in one of the problems. However, better results are obtained for the proposed reconfiguration mechanisms through anisotropic selection in terms of convergence time and hit rate for the benchmark problems tackled here. Roughly, it is also observed that constant and adaptive grid reconfiguration do not show an important difference in convergence time between them. Next, results for the real-world problems are presented.

Real-world Problems

In Table 4.3 the results for binary tournament selection are presented. The FMS

Table 4.2: Convergence time[1] and hit rate[2] for continuous problems. Local selection: central + anisotropic selection.

TC | RM [3]	Rastrigin	Griewank	Langerman
Square | None	230.38 ± 15.51 100%	444.73 ± 41.52 67%	163.61 ± 42.23 95%
Double | Constant	224.42 ± 19.22 100%	445.09 ± 37.19 68%	161.08 ± 49.35 96%
Double | Phenotypic	218.31 ± 20.56 100%	435.70 ± 31.28 79%	151.17 ± 37.01 98%
Double | Genotypic	216.53 ± 17.70 100%	433.58 ± 36.44 76%	159.95 ± 46.27 97%
ANOVA/K-W	(+)	(+)	•
Triple | Constant	217.90 ± 14.41 100%	440.85 ± 37.73 75%	147.90 ± 31.91 93%
Triple | Phenotypic	217.83 ± 19.52 100%	416.92 ± 41.24 83%	150.43 ± 43.03 96%
Triple | Genotypic	215.42 ± 17.63 100%	424.58 ± 45.50 78%	158.77 ± 36.78 94%
ANOVA/K-W	(+)	(+)	(+)
Quadruple | Constant	**188.91** ± 15.96 100%	**402.91** ± 47.46 93%	**117.90** ± 28.00 96%
Quadruple | Phenotypic	196.71 ± 29.43 100%	400.56 ± 45.63 91%	128.86 ± 26.68 97%
Quadruple | Genotypic	203.70 ± 18.51 100%	412.68 ± 43.95 83%	140.34 ± 38.07 97%
ANOVA/K-W	(+)	(+)	(+)

1 Convergence time is measured as the average number of generations for successful experiments. Corresponding standard deviation is included after ± symbol.

2 Hit rate is presented as the percentage of successful experiments out of the total number of experiments which is one hundred samples.

3 Topology Configuration (TC) refers to the number of configurations among which lattice reconfiguration is performed, see Subsection 4.1.3. Reconfiguration Mode (RM) refers to constant or adaptive lattice reconfiguration mechanisms.

problem is an epistatic, multi-modal and asymmetric problem. Its difficulty is reflected in the results where a low hit rate of less than 50% is obtained in all cGA reconfigurations cases. In [82] p. 42, for similar configuration constraints, except for the recombination method (double point crossover was employed) and the replacement policy, 27% hit rate is reported. Also in [82] p. 63, results applying binary tournament selection for both parents reported a hit rate of 63% for the best case scenario. On the other hand, the hit rate obtained for the FMS problem with a static square topology is 41% while constant reconfiguration alternatives present an increase of 2% to 5%. Similarly, adaptive reconfiguration improves from 1% to 8% for most cases, except for the quadruple genotype based reconfiguration mechanism where the hit rate is 33%. The convergence time is improved when applying the reconfiguration techniques in 10%. However, results do not provide statistical difference for any of the proposed mechanisms.

Analysing the SLE results, the loss of efficacy with respect to the number of successful experiments is noticeable: from 39% hit rate for the static square topology to 14% for the triple topology configuration. Although, there is one case, double topology case, presenting a better hit rate of 44%. In general the algorithm's hit rate is negatively affected when dynamic reconfiguration mechanisms are applied. In contrast, a slight improvement of 10% is obtained in terms of convergence time for the quadruple topology option with constant and phenotypic based reconfiguration.

For the GPS problem a smaller population size of 64 individuals is consistently used due to previous research reported in [89]. However, a more accurate threshold has been defined in the current experiments. Results do not show significant statistical difference as regards the convergence time and a slight improvement of 3% in terms of hit rate is obtained by the quadruple configuration case.

Table 4.4 shows the results for anisotropic selection. In contrast to binary tournament selection, the efficiency when solving the FMS problem is improved from 46% to 72% with quadruple constant based reconfiguration and in general all proposed mechanisms improve the hit rate. Similarly, a convergence time reduction of 25% is achieved. Yet, statistical significance is obtained for triple and quadruple topology configuration options. Comparable behaviour is shown by constant and adaptive mechanisms in the SLE problem: hit rate is improved from 19% to 40% when reconfiguration is performed while convergence time is reduced in approximately 15% with quadruple constant re-

Table 4.3: Convergence time[1] and hit rate[2] for real and combinatorial problems. Local selection: central + binary tournament.

TC | RM [3]	FMS	SLE	GPS	MMDP	MTTP
Square | None	210.78 ± 58.76 41%	297.28 ± 72.56 39%	73.12 ± 41.83 57%	431.31 ± 54.44 89%	259.65 ± 44.16 99%
Double | Constant	193.15 ± 75.23 46%	264.65 ± 50.01 33%	75.13 ± 44.50 59%	411.34 ± 70.79 89%	265.49 ± 54.48 99%
Double | Phenotypic	193.09 ± 74.05 45%	291.34 ± 122.03 29%	73.73 ± 41.05 57%	386.02 ± 51.66 85%	**239.45** ± 53.27 **100%**
Double | Genotypic	190.00 ± 57.57 44%	303.17 ± 100.88 44%	73.72 ± 65.39 48%	401.50 ± 61.56 86%	248.02 ± 49.86 100%
ANOVA/K-W	●	●	●	(+)	(+)
Triple | Constant	182.80 ± 37.44 46%	272.20 ± 74.59 29%	69.40 ± 36.07 47%	**385.16** ± 58.57 **92%**	266.90 ± 59.41 100%
Triple | Phenotypic	181.23 ± 46.15 42%	274.56 ± 97.75 15%	64.14 ± 30.37 57%	395.91 ± 76.57 90%	242.51 ± 46.60 100%
Triple | genotypic	191.42 ± 64.81 49%	289.45 ± 124.17 14%	75.19 ± 43.86 51%	398.10 ± 68.49 82%	248.39 ± 48.49 100%
ANOVA/K-W	●	●	●	(+)	(+)
Quadruple | Constant	189.78 ± 66.99 42%	253.05 ± 60.38 19%	64.66 ± 25.78 50%	361.71 ± 65.24 87%	252.16 ± 65.18 100%
Quadruple | Phenotypic	192.58 ± 56.96 41%	250.09 ± 55.55 21%	80.55 ± 54.250 60%	**384.38** ± 72.38 89%	248.46 ± 56.91 100%
Quadruple | Genotypic	185.21 ± 55.87 33%	299.33 ± 125.45 36%	69.67 ± 35.55 52%	387.02 ± 74.55 88%	259.00 ± 72.34 91%
ANOVA/K-W	●	(+)	●	(+)	●

1 Convergence time is measured as the average number of generations for successful experiments. Corresponding standard deviation is included after $\pm$ symbol.

2 Hit rate is presented as the percentage of successful experiments out of the total number of experiments which is one hundred samples.

3 Topology Configuration (TC) refers to the number of configurations among which lattice reconfiguration is performed, see Subsection 4.1.3. Reconfiguration Mode (RM) refers to constant or adaptive lattice reconfiguration mechanisms.

configuration. On the other hand, results for the GPS attitude determination problem show a hit rate improvement of 11%. However, the convergence time is maintained and in some cases is slightly increased by topology reconfiguration.

The difference in convergence time between binary tournament and anisotropic selection is significant. That difference is supported by the take-over times both methods present. Binary tournament is more exploitative in comparison to anisotropic selection, and therefore it implies shorter convergence times.

Combinatorial Problems

In Table 4.3, MMDP and MTTP results are included in the last two columns. For the MMDP, a size of 25 sub-problems has been evaluated, a maximum limit for the number of generations is increased to 700 (similarly to the SLE problem). On the other hand, for the MTTP a problem size of 100 tasks has been evaluated. In particular for this problem a population size of one hundred individuals has been employed. Several population sizes were initially tested and a population size of 100 individuals provided an adequate performance. However, it is not aimed in this study to determine the adequate population size as regards the size and difficulty of the problem but to evaluate the proposed mechanisms in order to improve the overall performance of cGAs.

The MMDP tackled through binary tournament selection provides a hit rate of 89%, similar efficacy is achieved by a couple of the proposed mechanisms presenting a slight improvement. The convergence time is reduced in 10%. Results also present statistical significance when compared to a static square topology. On the other hand, the hit rate is 100% in most test scenarios for the MTTP, while the convergence time is similar in all cases.

The results for anisotropic local selection are presented in Table 4.4. The highest hit rate for the MMDP is obtained with a static square topology (97%) or double constant reconfiguration (99%). It is noticeable that for the rest double and all triple reconfiguration mechanisms, the efficiency drops to 30% in the worst case scenario. However when quadruple constant or adaptive reconfiguration is applied, the hit rate recovers to 91%. The fourth topology configuration option provides in average the highest selective pressure which is probably the cause of that improvement. Comparing these results with those obtained through binary tournament selection which locally induces higher selection pressure, a minimum 80% hit rate is achieved in all experimental cases. There-

Table 4.4: Convergence time[1] and hit rate[2] for real and combinatorial problems. Local selection: central + anisotropic.

TC | RM [3]	FMS	SLE	GPS	MMDP	MTTP
Square | None	379.38 ± 67.83 46%	594.21 ± 54.23 19%	80.73 ± 20.36 76%	568.98 ± 54.01 97%	444.45 ± 36.92 31%
Double | Constant	361.26 ± 68.11 53%	542.98 ± 92.57 **40%**	95.45 ± 27.83 82%	$\mathbf{541.06} \pm 53.35$ **99%**	450.35 ± 30.11 44%
Double | Phenotypic	357.81 ± 64.74 54%	524.28 ± 86.57 32%	80.91 ± 29.42 81%	641.14 ± 34.43 50%	446.78 ± 33.38 50%
Double | Genotypic	346.80 ± 68.15 60%	542.07 ± 103.43 28%	86.77 ± 26.86 81%	651.86 ± 42.09 45%	436.22 ± 41.03 58%
ANOVA/K-W	•	(+)	(+)	(+)	•
Triple | Constant	356.12 ± 69.12 58%	551.05 ± 71.18 20%	95.96 ± 37.07 75%	660.26 ± 29.96 30%	446.55 ± 32.12 42%
Triple | Phenotypic	343.45 ± 64.35 **72%**	552.40 ± 112.65 37%	83.46 ± 32.89 79%	652.81 ± 34.99 49%	445.90 ± 38.72 60%
Triple | Genotypic	353.51 ± 74.59 68%	551.60 ± 91.13 33%	$\mathbf{77.40} \pm 17.59$ **85%**	638.38 ± 48.23 47%	446.93 ± 36.07 59%
ANOVA/K-W	(+)	(+)	(+)	(+)	•
Quadruple | Constant	$\mathbf{300.32} \pm 69.75$ 71%	$\mathbf{506.23} \pm 22.29$ 38%	86.76 ± 25.15 80%	558.56 ± 57.01 91%	$\mathbf{403.00} \pm 47.85$ **87%**
Quadruple | Phenotypic	321.91 ± 67.11 61%	526.63 ± 84.10 38%	81.75 ± 23.07 87%	608.13 ± 47.26 87%	428.17 ± 50.21 75%
Quadruple | Genotypic	323.13 ± 71.39 58%	529.20 ± 97.26 35%	79.79 ± 29.88 72%	597.54 ± 49.45 88%	431.04 ± 40.62 63%
ANOVA/K-W	(+)	(+)	•	(+)	(+)

1 Convergence time is measured as the average number of generations for successful experiments. Corresponding standard deviation is included after ± symbol.

2 Hit rate is presented as the percentage of successful experiments out of the total number of experiments which is one hundred samples.

3 Topology Configuration (TC) refers to the number of configurations among which lattice reconfiguration is performed, see Subsection 4.1.3. Reconfiguration Mode (RM) refers to constant or adaptive lattice reconfiguration mechanisms.

fore, to find the solution of the MMDP requires to sustain a more exploitative search during the evolutionary process to avoid stagnation. In terms of the average number of generations statistical significance is obtained in all experimental sets but it does not represent an improvement for the proposed lattice reconfiguration mechanisms.

For the MTTP through anisotropic selection, the hit rate is improved by all proposed lattice reconfiguration cases. From 31% without reconfiguration to 87% with quadruple constant lattice reconfiguration. The results do not have statistical significance in terms of convergence time in double and triple reconfiguration cases; contrary to the quadruple topology reconfiguration case where an improvement of 10% is achieved.

4.1.3.2 Summary

In this subsection, the research has focused on the effect of constantly or adaptively modifying the NGR through the lattice reconfiguration while internally maintaining toroidal like sub-structures. Providing different levels of selective pressure through several shapes and sizes in the internal sub-structures helps in promoting an adequate balance of the exploration-exploitation trade-off. Previously, researchers reported a better performance when the whole population shape was constantly or adaptively changed during evolution [6]. However, an implicit migration mechanism is induced and the effect of modifying the NGR while changing the whole population topology has not been separately evaluated. Here, individuals maintain their location with respect to the rest of the population while dynamically subdividing the entire population into smaller square, rectangular or linear sub-arrays.

Several problems have been tested in order to evaluate the proposed mechanisms from continuous to real and combinatorial problems. Emphasis has been put on evaluating the proposed mechanisms rather than the individual effect of genetic operations, thus maintaining cGAs operation close to their standard form. Two local selection methods presenting opposite selection pressures have been used in order to evaluate the flexibility introduced by combining higher or lower NGRs through the dynamic reconfiguration of the grid.

Binary tournament presents higher selection pressure and less flexibility when more than one topology configuration is used during evolution. In contrast, anisotropic selection provides lower selection pressure and better flexibility through the dynamic control of the NGR when the population topology is reconfigured. For binary tournament selection, most benchmark problems achieved a better hit rate while the convergence time was reduced in comparison to a cGA evolving on a static square topology. However, a better trade-off of both performance measures was obtained for intermediate cases such as having double and triple NGR alternatives. For anisotropic selection, having four topology configurations outperformed a static square topology in terms of convergence time consistently in most benchmark problems except for the MMDP and the GPS problem.

Comparing the dynamic lattice reconfiguration mechanisms, constant reconfiguration slightly outperformed adaptive approaches in several cases, a deeper analysis in this respect is included in the second experimental set-up. In the next section, diversity measures are refined in order to enhance the current adaptive algorithmic performance.

4.1.4 Second Experimental Set-Up: *Internal* vs *External* Lattice Reconfiguration

In the second experimental set-up, the main aim is to incorporate a direct comparison to the static, pre-programmed and adaptive criteria proposed by Dorronsoro et al., as well as to change the phenotypic diversity measure in order to improve the adaptive approach. Moreover, statistical analysis is extended to evaluate the algorithmic efficacy and a multiple comparison test for the convergence time is also carried out.

In Subsections 4.1.1 and 4.1.2, the expected effect of internally reconfigure the topology of the population is explained in detail. Together with three criteria to perform the grid reconfiguration: constantly or adaptively according to diversity measures at the phenotypic and genotypic spaces. In particular, the measurement of phenotypic diversity evaluated by Dorronsoro et al. in [6] which does not consider all possible scenarios is discarded and measuring the phenotypic entropy of the population every generation is proposed instead. See Subsection 4.1.2.2 for details of the new proposed measure.

As a consequence of the results obtained in the first experimental set up, having a quadruple topology option to reconfigure the grid provided the most distinctive performance results when analysing the effect of dynamically changing the topology of the population during evolution. In this subsection, only the quadruple topology configuration option is experimentally evaluated. Thus, the reconfiguration case drawn in Figure 4.1 is executed, constantly or adaptively.

Crossover and mutation operators are configured in a simple form and are kept constant during the search process. Single Point Crossover (SPC) operates pairs formed by a central individual and an individual selected from its local neighbourhood with probability $P_c = 1.0$. SPC randomly selects a single point for recombination. On the other hand, mutation is performed after recombination with probability $P_m = 0.02$. These genetic operators are not evaluated from a dynamic perspective.

As mentioned in previous chapters, replacement policies also play an important role in GAs. In cGAs, an offspring can replace a current individual in the following three ways: 1) always, 2) only if it has a better fitness score or 3) following some other condition. In this experimental set-up an *only replace if better* policy is followed. On the other hand, synchronous population updating is also applied. Synchronous updating requires current and next generations to be stored separately; once all individuals have evolved, new offspring will entirely replace the current population. Asynchronous updating has also been investigated and in several problems outperforms synchronous updating in terms of convergence time although the results accuracy and hit rate are negatively affected [42].

The following experimental constraints are evaluated:

- A population size of 400 individuals was used for most problems. Due to their size, GPS and MTTP problems are tackled using a population size of 64 and 100 individuals respectively.

- Local neighbourhood configuration is Von Neumann or linear composed by four individuals plus a central individual.

- One hundred independent runs are carried out per experimental reconfiguration criterion.

- A problem specific threshold based on the population's average fitness is used as a stop condition.

- A limit of 500 generations is used in most problems, with the exception of the Langerman function, the SLE and the MMDP problems with a limit of 700 generations.

The same reconfiguration mechanisms are applied to all problems and results are analysed based on convergence time and hit rate. The accuracy of results is determined by problem specific thresholds, see Appendix A. The stop condition evaluates the population average fitness score. Thus, the best individual solution has been spread through the entire grid. Commonly, researchers use as a stop condition a maximum limit in the number of generations or when an individual has reached the global optimum. It has also been established that experimental samples should evolve until a solution of the same quality has been found and not up to a limit in the number of generations [42]. The stop condition herein considered has been defined due to previous research constraints related to fault tolerance. Although fault tolerance is a different topic, outwith the scope of this subsection, the decision was made to keep the same stop condition for consistency.

In order to statistically support results so far obtained, the Lilliefors normality test, at 5% of significance, is performed on each set of convergence time results. This test is suitable when a fully-specified null distribution is unknown, in contrast to the Kolmorov-Smirnov test. Once normality of results has been established, an Analysis of Variance (ANOVA) is applied among results that follow a normal distribution whereas the Kruskal-Wallis test is applied to results that are not normally distributed. For sets of results with non normal distributions, dispersion of data is calculated using the *mean absolute deviation* and *italics* are used to highlight these cases in the results tables.

In the next subsection, results are summarized in two tables, one for static, pre-programmed and constant reconfiguration criteria and a second one for adaptive approaches. Then, in order to prove that a certain approach outperforms others, in terms of convergence time, a multiple comparison test is carried out, after a statistical analysis is done independently for each table of results. In contrast, if statistical proof is obtained only in one of the two tables, a multiple comparison test is applied separately.

In the results tables, where a statistically significant difference at 5% has been found among convergence time results, this has been represented by the symbol $+$. On the contrary, a $\bullet$ symbol represents results which are not statistically different, and where therefore the application of proposed dynamic criteria makes no difference in terms of the number of generations. On the other hand, having statistically different results does not mean the proposed approaches definitively improve cGAs performance. This statistical analysis only evaluates the efficiency of proposed techniques. Therefore, the efficacy of results has also been statistically considered.

The hit rate has been statistically evaluated as a Bernoulli trial. A random experiment whose result is either success or failure can be considered a Bernoulli trial [32]. For each dynamic lattice reconfiguration criterion, 100 experiments were carried out. In successful experiments, the global optima, at a certain threshold, is reached. Measuring the standard deviation of percentages of successful experiments provides a numerical value that indicates how significantly different are these hit rates. The standard deviation for each experimental sample is calculated as follows:

$$\sigma = \sqrt{r * \hat{p}\,(1 - \hat{p})} \tag{4.2}$$

where $\hat{p}$ represents the probability of successful experiments and r is the total number of experiments. In the next subsection, results are detailed.

4.1.4.1 Results Analysis

The results and corresponding analysis presented in this subsection, aims to empirically support the main hypothesis of this study: dynamic modification of the NGR through the lattice reconfiguration would lead to significant improvements in the performance of cGAs. At a local level, as explained previously in Section 4.1.1, two different local selection methods have been applied. The main difference between them consists in applying higher (binary tournament) or lower (anisotropic) selection pressure. These two methods were chosen because of their considerably different selection intensity. Thus, it is possible to separately evaluate 1) the effect of having a highly exploitative or explorative cellular GA, due to local selection, and 2) the added flexibility of dynamically reconfiguring the population topology in order to supply the search process with a more convenient and balanced exploitation-exploration trade-off.

For presentation, results are divided by local selection method. Two tables of results are organized: the first includes static, pre-programmed (*external*) reconfiguration and constant (*internal*) reconfiguration. The second group presents adaptive approaches for both *external* and *internal* lattice reconfiguration approaches.

In the tables, numbers in **bold** indicate the best average convergence time and *italics* are used to distinguish between standard deviation (normal distributions) and mean absolute deviation (non normal distributions) as dispersion measures. Next to the hit rates, corresponding Bernoulli trial standard deviations are shown in small numbers.

Binary Tournament Local Selection

Continuous problems corresponding to well known mathematical functions (Rast-

Table 4.5: Convergence time[1] and hit rate[2] for continuous and the FMS problems through static, preprogrammed (external) and constant (internal) lattice reconfiguration with binary tournament local selection

Static/Preprog.				
External Reconfig.	Rastrigin	Griewank	Langerman	FMS
Square	*136.68* ± *9.48*	272.79 ± 46.45	291.04 ± 103.92	210.78 ± 58.76
	100%	89%, $_{3.12}$	21%, $_{4.07}$	41%, $_{4.91}$
Rectangular	148.01 ± *11.49*	315.41 ± 63.12	285.52 ± *98.34*	247.70 ± 80.70
	100%	90%, $_{3}$	19%, $_{3.92}$	**48%**, $_{4.99}$
Narrow	219.12 ± *22.56*	412.74 ± 45.09	417.33 ± 71.00	406.52 ± 64.75
	100%	63%, $_{4.82}$	3%, $_{1.70}$	34%, $_{4.73}$
Square-Narrow	150.23 ± *21.96*	313.48 ± 83.23	226.31 ± *76.08*	206.18 ± *46.72*
	37%, $_{4.82}$	**100%**	19%, $_{4.20}$	19%, $_{3.92}$
Narrow-Square	139.26 ± 13.09	262.08 ± *34.10*	305.35 ± 100.08	207.42 ± 40.20
	100%	96%, $_{1.95}$	17%, $_{3.75}$	35%, $_{4.76}$
Constant				
Internal Reconfig.				
Square→Constant	**122.79** ± *8.33*	**245.35** ± *43.90*	318.73 ± *164.18*	195.23 ± *54.12*
	100%	80%, $_{4.00}$	**23%**, $_{4.20}$	43%, $_{4.95}$
$4(\frac{n}{2} \times \frac{n}{2})$→Constant	123.09 ± *9.84*	245.86 ± *38.15*	**222.90** ± *88.46*	**189.78** ± 46.32
	100%	73%, $_{4.43}$	20%, $_{4.00}$	42%, $_{4.93}$
ANOVA/K-W [3]	+	+	●	●

1 Convergence time is measured as the average number of generations for successful experiments. Corresponding standard deviation is included after ± symbol.

2 Hit rate is presented as the percentage of successful experiments out of the total number of experiments which is one hundred samples.

3 Statistical tests: (+) statistical difference is proved / ● is not proved.

Table 4.6: Convergence time[1] and hit rate[2] for continuous and the FMS problems through external and internal adaptive lattice reconfiguration with binary tournament local selection

Adaptive Phenotypic				
External Reconfig.	Rastrigin	Griewank	Langerman	FMS
Rectangular→Adapt.	135.31 ± *8.45* 100%	263.63 ± 39.72 **88%**, $_{3.24}$	261.27 ± 106.77 **29%**, $_{4.53}$	**178.60** ± *29.73* **53%**, $_{4.99}$
Narrow→Adapt.	137.00 ± *10.20* 100%	260.94 ± 43.13 84%, $_{3.66}$	**218.20** ± *62.26* 24%, $_{4.27}$	185.44 ± 35.87 47%, $_{4.99}$
Internal Reconfig.				
Square→Adapt.	126.25 ± 12.05 100%	254.67 ± 41.94 77%, $_{4.20}$	223.76 ± *80.54* 17%, $_{3.75}$	203.27 ± *59.78* 48%, $_{4.99}$
$4(\frac{n}{2} \times \frac{n}{2})$→Adapt.	129.75 ± *10.69* 100%	255.58 ± *38.74* 68%, $_{4.66}$	224.60 ± 105.55 15%, $_{3.57}$	188.17 ± *48.32* 47%, $_{4.99}$
Adaptive Genotypic				
External Reconfig.				
Rectangular→Adapt.	126.66 ± *8.33* 100%	**250.05** ± *39.85* 76%, $_{4.27}$	276.94 ± *111.51* 19%, $_{3.92}$	167.39 ± 26.65 46%, $_{4.98}$
Narrow→Adapt.	**125.77** ± *8.42* 100%	253.73 ± 38.81 76%, $_{4.27}$	237.55 ± *72.31* 20%, $_{4.00}$	179.95 ± 33.41 49%, $_{4.99}$
Internal Reconfig.				
Square→Adapt.	127.16 ± *9.38* 100%	260.62 ± *45.90* 77%, $_{4.20}$	239.78 ± 107.84 14%, $_{3.46}$	213.88 ± *65.61* 34%, $_{4.73}$
$4(\frac{n}{2} \times \frac{n}{2})$→Adapt.	126.28 ± *8.51* 100%	255.14 ± *39.52* 81%, $_{3.92}$	261.38 ± *101.47* 18%, $_{3.84}$	195.71 ± 57.06 42%, $_{4.93}$
ANOVA/K-W[3]	+	●	●	●

1 Convergence time is measured as the average number of generations for successful experiments. Corresponding standard deviation is included after ± symbol.

2 Hit rate is presented as the percentage of successful experiments out of the total number of experiments which is one hundred samples.

3 Statistical tests: (+) statistical difference is proved / ● is not proved.

rigin, Griewank and Langerman functions) are tackled considering an average fitness score threshold in the order of $\bar{f} \approx 1 \times 10^{-4}$. On the other hand, although real chromosomes encoding might be more appropriate to solve these kind of problems [11], it has been decided to explore proposed cellular configurations using binary encoding. The main reason is that the authors' research group investigates the design and implementation of reconfigurable architectures targeting hard real-time applications [53, 51]. Hence, using binary encoding in this research allows the extension of current cGAs research to the actual implementation of these techniques on reconfigurable fabrics in which resources for real data processing and representation are strictly limited. For more details on problem representation for evolutionary techniques, readers are referred to [7].

Tables 4.5 and 4.6 present results for static, pre-programmed, constant and adaptive lattice reconfiguration criteria through binary tournament local selection. The Rastrigin function hit rate is 100% for all configurations, and convergence time is slightly affected through lattice reconfiguration. Statistical proof is obtained for the best criterion: square $\rightarrow$ constant *internal* reconfiguration. On the other hand, adaptive approaches do not show significant difference in convergence time.

Similar results in the number of generations are obtained for the Griewank function, statistical difference at 5% is proved for square $\rightarrow$ constant *internal* reconfiguration with respect to static and pre-programmed criteria. However, in terms of hit rate, pre-programmed square $\rightarrow$ narrow approach shows major advantage in this regard, from 80% ($\sigma = 4.0$) to 100% $\sigma = 0.0$, based on calculated sample standard deviations. On the other hand, for adaptive approaches there is no significant difference amongst them in terms of convergence time. The best hit rate is 88% ($\sigma = 3.2496$) for rectangular $\rightarrow$ adaptive *external* reconfiguration, which is not as good as the pre-programmed approach.

The Langerman function is the most difficult in the continuous problems set; hit rates are not higher than 29%. The improvement for lattice reconfiguration approaches is not significant, either in convergence time or in the hit rate for all proposed criteria through binary tournament local selection. Similar results were obtained for the FMS problem. No significant improvement in terms of convergence time was obtained through any of the proposed approaches. Standard deviations for hit rates do not show significant statistical difference either.

From these two sets of results, it is worth noticing that the best convergence times in Table 4.5 correspond to constant approaches. However, hit rates in some cases are higher for either static or pre-programmed criteria, with statistical proof, for example, in the Griewank function. For adaptive approaches, in Table 4.6, best convergence times correspond to *external* reconfiguration techniques, however these results are not statistically supported. Results consistency for convergence times is maintained for most problems, except in the Langerman function.

In Tables 4.7 and 4.8, corresponding results for SLE, GPS and combinatorial problems, through binary tournament selection, are shown. In terms of convergence time, the SLE problem presents a significant difference for square $\rightarrow$ constant *internal* reconfiguration. However, in terms of hit rate, a standard static square grid provides the highest efficacy (39%). In contrast, the best phenotypic adaptive ($4(\frac{n}{2} \times \frac{n}{2}) \rightarrow$ adaptive) and genotypic (square $\rightarrow$ adaptive) adaptive criteria achieve a hit rate of 30%, with a minimum significant difference due to $\sigma = 4.8774$ (39%) and $\sigma = 4.5825$ (30%) respectively. The convergence times for adaptive criteria are not different, and the consistency of the results is not well maintained.

For the GPS attitude determination problem, through binary local selection, no statistical proof was obtained in terms of convergence time and hit rate for both sets of results. In contrast, the MMDP statistical proof was found in convergence time for $4(\frac{n}{2} \times \frac{n}{2}) \rightarrow$ constant *internal* reconfiguration. However, the best hit rate of 92% ($\sigma = 2.7129$) is reached through pre-programmed narrow $\rightarrow$ square *external* reconfiguration, in comparison to 87% ($\sigma = 3.363$) of former approach. However, the narrow $\rightarrow$ adaptive genotypic based approach, *external* reconfiguration, provides the best efficiency and efficacy.

For the MTTP, high hit rates are in general obtained. There is a statistical difference for the best configuration case among static, pre-programmed and constant approaches: square $\rightarrow$ constant *internal* reconfiguration. Similarly for adaptive approaches, statistical difference is proved for the best adaptive mechanism: narrow $\rightarrow$ adaptive genotypic based *external* reconfiguration. After a multiple comparison test is performed for both sets of results, no statistical proof was found for the adaptive approach in terms of convergence time.

Finally, for the P-Peaks problem through binary tournament selection, no statistical difference was obtained for either static, pre-programmed, constant or for adaptive approaches in terms of convergence time. On the other hand, the hit rates were in all cases 100%.

Anisotropic Local Selection

Results obtained through anisotropic local selection are presented in this subsection. In Tables 4.9 and 4.10 results for continuous problems and the FMS problem are presented. Anisotropic local selection with constant $\alpha = 0.8$ presents a more explorative behaviour if compared to binary tournament selection. Moreover, this selection method also makes use of structural properties in cGAs, due to the neighbourhood configuration in which the location of individuals for selection is implied.

In both tables, statistical difference is proved for all problems. Therefore, multiple comparison tests were carried out among all approaches. The Rastrigin function shows statistical difference in convergence time for both sets of results. However, no statistical proof supports this difference between best approaches on each table. In comparison to a static square topology, convergence time consistency is held in constant and adaptive criteria.

The Griewank function is better approached by *internal* reconfiguration, square $\rightarrow$ constant for best convergence time, and $4(\frac{n}{2} \times \frac{n}{2}) \rightarrow$ constant for best hit rate. Similar results are obtained through adaptive approaches. Genotypic based *internal* reconfiguration also provides the best performance. In terms of the hit rate, 95% and 93% are reached respectively. However, after multiple comparison, no statistical proof was obtained between constant and adaptive criteria. Moreover, consistency in convergence time is kept through *internal* and adaptive *external* mechanisms.

The Langerman function also provides the best performance through constant and phenotypic *internal* reconfiguration. Similar to the previous function, statistical proof, in convergence time, was not found between the best constant and adaptive approaches. On the other hand, convergence time consistency is slightly lost through *internal* constant and adaptive reconfiguration. In terms of hit rate, square $\rightarrow$ constant and $4(\frac{n}{2} \times \frac{n}{2})$ $\rightarrow$ adaptive achieve up to 37% and 32% hit rate respectively. This represents a significant improvement in efficacy in comparison to a static square topology, from $\sigma = 4.074$ to $\sigma = 4.8280$ and $\sigma = 4.6647$ in standard deviations.

The last problem presented in Tables 4.9 and 4.10 is the real FMS problem. This problem is highly epistatic, which means genetic interdependency is strong. Constant

Table 4.7: Convergence time[1] and hit rate[2] for real and combinatorial problems through static, preprogrammed (external) and constant (internal) lattice reconfiguration with binary tournament local selection

Static/Preprog. *External* Reconfig.	SLE	GPS	MMDP	MTTP	P-Peaks
Square	297.28 ± 72.56 **39%**, $_{4.87}$	73.12 ± *28.69* 57%, $_{4.95}$	431.31 ± *38.45* 89%, $_{3.12}$	259.65 ± 44.16 99%, $_{0.99}$	**169.46** ± *19.99* 100%,
Rectangular	359.09 ± *87.22* 31%, $_{4.62}$	85.24 ± *30.04* 53%, $_{4.99}$	517.14 ± 63.87 90%, $_{3.00}$	295.57 ± 48.70 99%, $_{0.99}$	175.52 ± *23.11* 100%,
Narrow	573.25 ± 67.44 16%, $_{3.66}$	123.57 ± 43.77 **66%**, $_{4.73}$	696.00 ± 0.00 1%, $_{0.99}$	− − −− −−	170.34 ± *23.36* 100%,
Square-Narrow	347.55 ± *119.30* 29%, $_{4.53}$	98.84 ± *45.76* 46%, $_{4.98}$	455.56 ± 84.93 86%, $_{3.46}$	314.85 ± 69.92 97%, $_{1.70}$	209.58 ± *30.71* 100%,
Narrow-Square	298.66 ± 89.93 27%, $_{4.43}$	85.62 ± *32.19* 58%, $_{4.93}$	488.57 ± 51.19 **92%**, $_{2.71}$	276.41 ± 48.50 99%, $_{0.99}$	242.25 ± 44.63 100%,
Constant *Internal* Reconfig.					
Square→Constant	**257.09** ± *77.29* 21%, $_{4.07}$	72.55 ± *30.18* 54%, $_{4.98}$	358.03 ± *41.76* 88%, $_{3.24}$	**243.08** ± *35.77* 98%, $_{1.70}$	170.68 ± *21.00* 100%,
$4(\frac{n}{2} \times \frac{n}{2})$→Constant	269.75 ± *79.87* 28%, $_{4.49}$	**71.00** ± *20.22* 53%, $_{4.99}$	**352.72** ± *54.93* 87%, $_{3.36}$	252.16 ± *43.58* **100%**,	173.99 ± *25.04* 100%,
ANOVA/K-W [3]	+	●	+	+	●

1 Convergence time is measured as the average number of generations for successful experiments. Corresponding standard deviation is included after ± symbol.

2 Hit rate is presented as the percentage of successful experiments out of the total number of experiments which is one hundred samples.

3 Statistical tests: (+) statistical difference is proved / ● is not proved.

Table 4.8: Convergence time[1] and hit rate[2] for real and combinatorial problems through external and internal adaptive lattice reconfiguration with binary tournament local selection

Adaptive Phenotypic					
External Reconfig.	SLE	GPS	MMDP	MTTP	P-Peaks
Rectangular→Adapt.	304.72 ± *83.51* 29%, 4.53	68.78 ± *29.60* 60%, 4.89	411.24 ± *42.25* 90%, 3.00	231.58 ± *34.06* 100%,	180.50 ± *24.84* 100%,
Narrow→Adapt.	298.76 ± 104.25 26%, 4.33	65.58 ± *20.44* 53%, 4.99	427.91 ± 57.66 89%, 3.12	226.52 ± 36.68 100%,	181.51 ± *25.91* 100%,
Internal Reconfig.					
Square→Adapt.	277.05 ± 84.93 20%, 4.00	78.17 ± *31.09* 51%, 4.99	401.39 ± *56.07* 93%, 2.55	246.40 ± *34.50* 100%,	180.46 ± *26.39* 100%,
$4(\frac{n}{2} \times \frac{n}{2})$→Adapt.	290.60 ± *88.61* **30%**, 4.58	70.17 ± *24.16* **63%**, 4.82	385.97 ± *51.20* 88%, 3.24	256.14 ± *41.47* 99%, 0.99	177.64 ± *27.40* 100%,
Adaptive Genotypic					
External Reconfig.					
Rectangular→Adapt.	**232.50** ± 70.52 26%, 4.38	**63.00** ± *22.51* 58%, 4.93	366.14 ± *43.70* 91%, 2.86	227.22 ± *32.77* 100%,	177.55 ± *23.93* 100%,
Narrow→Adapt.	274.73 ± 90.69 23%, 4.20	73.12 ± *31.02* 56%, 4.96	**360.77** ± 40.39 **94%**, 2.37	**222.55** ± *31.01* 99%, 0.99	**166.65** ± 22.66 100%
Internal Reconfig.					
Square→Adapt.	269.86 ± *64.49* **30%**, 4.58	65.36 ± *20.60* 49%, 4.99	388.94 ± *43.06* 89%, 3.12	254.06 ± *38.35* 100%,	174.66 ± *26.20* 100%,
$4(\frac{n}{2} \times \frac{n}{2})$→Adapt.	272.84 ± 117.49 25%, 4.33	72.03 ± *30.39* 52%, 4.99	397.88 ± *49.58* 90%, 3.00	252.81 ± *39.91* 100%,	174.98 ± *28.53* 100%,
ANOVA/K-W [3]	●	●	+	+	●

1 Convergence time is measured as the average number of generations for successful experiments. Corresponding standard deviation is included after ± symbol.

2 Hit rate is presented as the percentage of successful experiments out of the total number of experiments which is one hundred samples.

3 Statistical tests: (+) statistical difference is proved / ● is not proved.

Table 4.9: Convergence time[1] and hit rate[2] for continuous and the FMS problems through static, preprogrammed (external) and constant (internal) lattice reconfiguration with anisotropic local selection

Static/Preprog. *External* Reconfig.	Rastrigin	Griewank	Langerman	FMS
Square	230.38 ± 15.51 100%	438.36 ± 39.04 68%, $_{4.66}$	432.61 ± 47.01 21%, $_{4.07}$	$379.38 \pm \mathit{58.58}$ 52%, $_{4.99}$
Rectangular	283.34 ± 27.42 100%	464.18 ± 23.06 16%, $_{3.66}$	482.00 ± 0.00 1%, $_{0.99}$	429.35 ± 61.08 14%, $_{3.46}$
Narrow	$470.77 \pm \mathit{19.00}$ 53%	− − −− −−	− − −− −−	− − −− −−
Square-Narrow	$279.35 \pm \mathit{29.43}$ 100%	437.57 ± 52.99 14%, $_{3.46}$	430.50 ± 34.54 6%, $_{2.37}$	368.39 ± 75.91 33%, $_{4.70}$
Narrow-Square	246.43 ± 14.76 100%	431.53 ± 36.83 93%, $_{2.55}$	448.11 ± 36.08 18%, $_{3.84}$	420.51 ± 53.73 35%, $_{4.76}$
Constant *Internal* Reconfig.				
Square→Constant	$186.89 \pm \mathit{14.56}$ 100%	$\mathbf{370.11} \pm 45.66$ 89%, $_{3.12}$	316.62 ± 66.14 **37%**, $_{4.82}$	$305.24 \pm \mathit{61.08}$ 66%, $_{4.73}$
$4(\frac{n}{2} \times \frac{n}{2})$→Constant	$\mathbf{186.67} \pm \mathit{11.48}$ 100%	375.97 ± 45.76 **95%**, $_{2.17}$	$\mathbf{309.42} \pm \mathit{69.13}$ 26%, $_{4.38}$	$\mathbf{300.32} \pm \mathit{53.09}$ **71%**, $_{4.53}$
ANOVA/K-W [3]	+	+	+	+

1 Convergence time is measured as the average number of generations for successful experiments. Corresponding standard deviation is included after ± symbol.

2 Hit rate is presented as the percentage of successful experiments out of the total number of experiments which is one hundred samples.

3 Statistical tests: (+) statistical difference is proved / ● is not proved.

Table 4.10: Convergence time[1] and hit rate[2] for continuous and the FMS problems through external and internal adaptive lattice reconfiguration with anisotropic local selection

Adaptive Phenotypic				
External Reconfig.	Rastrigin	Griewank	Langerman	FMS
Rectangular→Adapt.	227.12 ± 17.92 100%	420.29 ± 40.40 86%, $_{3.46}$	420.21 ± 47.44 28%, $_{4.48}$	369.57 ± 64.78 61%, $_{4.87}$
Narrow→Adapt.	227.13 ± 15.99 100%	425.08 ± 40.74 84%, $_{3.66}$	$391.00 \pm \mathit{41.28}$ 25%, $_{4.33}$	358.77 ± 57.90 57%, $_{4.95}$
Internal Reconfig.				
Square→Adapt.	$\mathbf{198.14} \pm \mathit{14.03}$ 100%	391.32 ± 48.94 90%, $_{3.00}$	365.42 ± 79.82 19%, $_{3.92}$	$\mathbf{307.37} \pm \mathit{58.32}$ 61%, $_{4.87}$
$4(\frac{n}{2} \times \frac{n}{2})$→Adapt.	$200.01 \pm \mathit{12.89}$ 100%	392.80 ± 49.80 92%, $_{2.71}$	$\mathbf{345.21} \pm 67.97$ **32%**, $_{4.66}$	$308.82 \pm \mathit{55.23}$ 62%, $_{4.85}$
Adaptive Genotypic				
External Reconfig.				
Rectangular→Adapt.	202.39 ± 18.42 100%	401.90 ± 44.85 83%, $_{3.75}$	387.22 ± 59.31 27%, $_{4.43}$	328.87 ± 65.23 63%, $_{4.82}$
Narrow→Adapt.	$202.11 \pm \mathit{16.62}$ 100%	393.69 ± 49.50 81%, $_{3.92}$	382.73 ± 61.50 26%, $_{4.38}$	332.49 ± 70.22 59%, $_{4.91}$
Internal Reconfig.				
Square→Adapt.	$199.79 \pm \mathit{12.54}$ 100%	$\mathbf{388.31} \pm 45.69$ 91%, $_{2.86}$	357.40 ± 70.93 25%, $_{4.33}$	329.87 ± 75.44 66%, $_{4.73}$
$4(\frac{n}{2} \times \frac{n}{2})$→Adapt.	199.57 ± 14.48 100%	393.90 ± 44.62 **93%**, $_{2.55}$	348.50 ± 60.90 22%, $_{4.14}$	310.97 ± 59.45 **71%**, $_{4.53}$
ANOVA/K-W [3]	+	+	+	+

1 Convergence time is measured as the average number of generations for successful experiments. Corresponding standard deviation is included after ± symbol.

2 Hit rate is presented as the percentage of successful experiments out of the total number of experiments which is one hundred samples.

3 Statistical tests: (+) statistical difference is proved / ● is not proved.

internal reconfiguration gives the best performance through $4(\frac{n}{2} \times \frac{n}{2}) \rightarrow$ constant mechanism with statistical significant difference. On the other hand, adaptive phenotypic and genotypic *internal* reconfiguration mechanisms provide the best convergence time and hit rate respectively. However, once again no statistical proof was obtained between constant and adaptive reconfiguration techniques. Moreover, best hit rates are 71% in both cases. Consistency in convergence time is maintained among reconfiguration approaches.

It is worth noticing that through static rectangular topologies most of the problems in Table 4.9 lose performance quality, increasing convergence time and limiting their hit rates to the minimum, except in the Rastrigin function. Moreover, for static narrow lattices, three of the problems do not converge at all. An important observation is that the best performance results are obtained through constant and adaptive *internal* reconfiguration mechanisms.

In Tables 4.11 and 4.12, results for SLE, GPS and all combinatorial problems are presented. For most of the problems in each table, statistical difference was found in terms of convergence times, except for the MMDP problem in Table 4.11 and the SLE problem in Table 4.12.

The best convergence time for the SLE problem is achieved through $4(\frac{n}{2} \times \frac{n}{2}) \rightarrow$ constant *internal* reconfiguration together with the highest hit rate of 42%. Statistical difference is proved in the average number of generations with respect to a static square topology, as well as in hit rates where the difference in standard deviation ranges from $\sigma = 3.923$ to $\sigma = 4.9356$. On the other hand, dispersion of data increases through constant reconfiguration. In contrast, results for adaptive approaches are not significantly different in convergence time and best hit rate of 40% is achieved through genotypic based *external* reconfiguration. Nevertheless, convergence times are similar among adaptive mechanisms.

Overall, the GPS problem is the smallest problem. Yet, its landscape is multimodal and non symmetric also presenting high epistasis. A smaller population size was implemented with 64 individuals and for anisotropic selection, statistical proof was obtained in convergence time. Overall, the best convergence time is achieved through adaptive genotypic based *external* reconfiguration, although a substantially better hit rate of 91% is reached through a static narrow topology. On the other hand, compared with a static square topology, consistency in convergence time is preserved through constant and adaptive *internal* approaches.

The last three problems are combinatorial. Hit rates of 97% are achieved through static square and rectangular topologies for the MMDP problem, while a narrow topology cannot converge at all. No statistical proof is obtained in terms of convergence time. For adaptive approaches, the best performance was obtained through phenotypic and genotypic based $4(\frac{n}{2} \times \frac{n}{2}) \rightarrow$ adaptive *internal* reconfiguration with statistical difference in convergence time and 96% hit rate. After a multiple comparison test, statistical proof is also found between best convergence times on each table while consistency is also maintained. However, it is clear that a static square or rectangular topology provides the best overall performance.

The MTTP presents a very low hit rate for a static square population topology. Moreover, through rectangular and narrow topologies, cGAs are not at all appropriate to solve this problem. However, the best convergence time and hit rate are achieved by constant *internal* reconfiguration and the hit rate is improved up to 100%. Adaptive approaches show, with statistical difference, improvement in terms of convergence time. Narrow $\rightarrow$ adaptive genotypic based *external* reconfiguration outperforms other approaches. Yet, hit rate reaches 98% and 99% in the best cases. Convergence time consistency is kept among *internal* constant and adaptive reconfiguration.

Overall, hit rates for the P-Peaks problem are 100%. In terms of convergence time, statistical tests prove a significant improvement. Constant *internal* and adaptive genotypic based *external* reconfiguration outperform the rest of the proposed techniques, although, there is no statistical proof to differenciate between both approaches. The results consistency is well maintained through the best approaches.

Similar to the results in Table 4.9 for static, pre-programmed and constant approaches; in Table 4.11, the best convergence times and most of the best hit rates correspond to constant *internal* reconfiguration. Having a static topology or to perform a pre-programmed change in population topology during evolution does not provide the best performance cGAs could achieve. However, in some cases better hit rates are obtained through static and pre-programmed reconfiguration approaches.

To summarize this subsection, binary tournament and anisotropic local selection methods present distinctively different selection intensities. Binary tournament is more exploitative in comparison to anisotropic selection which configured with an $\alpha = 0.8$, slows down by three times the growth rate for the best individual. Therefore, con-

Table 4.11: Convergence time[1] and hit rate[2] for real and combinatorial problems through static, preprogrammed (external) and constant (internal) lattice reconfiguration with anisotropic local selection

Static/Preprog. *External* Reconfig.	SLE	GPS	MMDP	MTTP	P-Peaks
Square	594.21 ± 54.32 19%, 3.92	$80.73 \pm \mathit{15.43}$ 76%, 4.27	568.98 ± 54.01 **97%**, 1.70	444.45 ± 36.92 31%, 4.62	286.19 ± 35.95 99%
Rectangular	603.66 ± 54.68 6%, 2.37	$148.95 \pm \mathit{35.42}$ 88%, 3.24	574.03 ± 47.93 **97%**, 1.70	– – –– ––	274.80 ± 34.54 100%
Narrow	– – –– ––	$143.49 \pm \mathit{35.20}$ **91%**, 2.86	– – –– ––	– – –– ––	$304.96 \pm \mathit{37.53}$ 100%
Square-Narrow	585.87 ± 115.39 8%, 2.71	$93.69 \pm \mathit{31.25}$ 82%, 3.84	572.75 ± 61.37 77%, 4.20	462.00 ± 0.00 1%, 0.99	$321.44 \pm \mathit{38.42}$ 100%
Narrow-Square	600.17 ± 79.09 29%, 4.53	$84.23 \pm \mathit{20.01}$ 78%, 4.14	$633.40 \pm \mathit{31.46}$ 80%, 4.00	463.90 ± 19.46 11%, 3.12	349.29 ± 36.80 100%
Constant *Internal* Reconfig.					
Square→Constant	469.50 ± 114.32 42%, 4.93	$\mathbf{72.76} \pm \mathit{16.93}$ 78%, 4.14	$\mathbf{550.86} \pm 55.63$ 94%, 2.37	$437.34 \pm \mathit{65.97}$ **100%**	$\mathbf{256.77} \pm \mathit{28.67}$ 100%
$4(\frac{n}{2} \times \frac{n}{2})$→Constant	$\mathbf{442.80} \pm 93.42$ **42%**, 4.93	90.00 ± 23.70 89%, 3.12	553.96 ± 51.20 91%, 2.86	$\mathbf{403.00} \pm 47.85$ 87%, 3.36	$261.31 \pm \mathit{28.61}$ 100%
ANOVA/K-W [3]	+	+	●	+	+

1 Convergence time is measured as the average number of generations for successful experiments. Corresponding standard deviation is included after ± symbol.

2 Hit rate is presented as the percentage of successful experiments out of the total number of experiments which is one hundred samples.

3 Statistical tests: (+) statistical difference is proved / ● is not proved.

Table 4.12: Convergence time[1] and hit rate[2] for real and combinatorial problems through external and internal adaptive lattice reconfiguration with anisotropic local selection

Adaptive Phenotypic					
External Reconfig.	SLE	GPS	MMDP	MTTP	P-Peaks
Rectangular→Adapt.	569.00 ± 81.42 31%, $_{4.62}$	$75.96 \pm \mathit{12.25}$ 81%, $_{3.92}$	648.81 ± 35.58 33%, $_{4.70}$	377.73 ± 47.20 95%, $_{2.17}$	$290.93 \pm \mathit{35.31}$ 100%
Narrow→Adapt.	566.20 ± 90.81 25%, $_{4.33}$	80.03 ± 18.34 79%, $_{4.07}$	$659.28 \pm \mathit{22.11}$ 50%, $_{5.00}$	377.67 ± 38.69 97%, $_{1.70}$	$282.52 \pm \mathit{37.87}$ 100%
Internal Reconfig.					
Square→Adapt.	$\mathbf{482.66} \pm \mathit{95.83}$ 33%, $_{4.70}$	$82.39 \pm \mathit{20.51}$ 78%, $_{4.14}$	602.77 ± 52.03 85%, $_{3.57}$	429.14 ± 46.50 81%, $_{3.92}$	277.00 ± 46.50 100%,
$4(\frac{n}{2} \times \frac{n}{2})$→Adapt.	531.71 ± 106.29 38%, $_{4.85}$	$86.02 \pm \mathit{21.34}$ 72%, $_{4.48}$	601.77 ± 51.54 **96%**, $_{1.95}$	$424.02 \pm \mathit{35.34}$ 77%, $_{4.20}$	277.11 ± 41.62 100%,
Adaptive Genotypic					
External Reconfig.					
Rectangular→Adapt.	513.23 ± 90.67 30%, $_{4.58}$	$71.42 \pm \mathit{15.30}$ 82%, $_{3.84}$	618.30 ± 47.03 78%, $_{4.14}$	362.03 ± 42.30 **99%**, $_{0.99}$	$\mathbf{239.70} \pm \mathit{28.24}$ 100%
Narrow→Adapt.	525.77 ± 85.77 **40%**, $_{4.89}$	$\mathbf{65.75} \pm \mathit{12.93}$ 78%, $_{4.14}$	623.41 ± 48.18 77%, $_{4.20}$	$\mathbf{354.02} \pm 47.43$ 98%, $_{1.40}$	242.20 ± 39.50 100%
Internal Reconfig.					
Square→Adapt.	511.00 ± 118.66 30%, $_{4.58}$	$86.01 \pm \mathit{22.22}$ **84%**, $_{3.66}$	599.05 ± 47.81 92%, $_{2.71}$	432.82 ± 37.70 68%, $_{4.66}$	$268.78 \pm \mathit{32.56}$ 100%,
$4(\frac{n}{2} \times \frac{n}{2})$→Adapt.	515.55 ± 94.51 38%, $_{4.85}$	75.86 ± 16.19 75%, $_{4.33}$	$\mathbf{595.04} \pm 47.93$ 87%, $_{3.36}$	412.03 ± 44.29 77%, $_{4.20}$	262.84 ± 26.26 100%,
ANOVA/K-W [3]	●	+	+	+	+

1 Convergence time is measured as the average number of generations for successful experiments. Corresponding standard deviation is included after ± symbol.

2 Hit rate is presented as the percentage of successful experiments out of the total number of experiments which is one hundred samples.

3 Statistical tests: (+) statistical difference is proved / ● is not proved.

vergence times are not directly comparable. In most problems, the average number of generations is duplicated through anisotropic selection. Hence, results analysis has been carried out separately for each local selection method.

Broadly speaking, Tables 4.5, 4.7, 4.9 and 4.11 show a persistent tendency for constant *internal* reconfiguration achieving better convergence times. This is not fully repeated in terms of hit rate, which in some cases is improved through static and pre-programmed approaches.

For adaptive approaches, results do not show a predominant improvement through any of the proposed approaches. Through phenotypic diversity measures, *external* reconfiguration calculates the average fitness score achieved in each generation. Rather, through *internal* reconfiguration, phenotypic diversity is monitored through its entropy. On the other hand, genotypic diversity is in both, *external* and *internal* cases, measured through its entropy, which for binary chromosomes encoding is determined using the Hamming distance. Finally and most importantly, in just a few cases, statistical proof was obtained when comparing constant *internal* reconfiguration cases and the best adaptive approaches. Therefore, constant reconfiguration results in a better approach to improving cGAs performance through dynamically modifying the topology of the population and is also computationally less expensive.

4.1.4.2 Summary

The aim of the second experimental set-up has been to investigate how structural properties in cGAs can be a factor in adequately tuning diversity during evolution. *Internal* lattice reconfiguration had been experimentally tested and compared against *external* reconfiguration proposed in [6]. The main difference of both approaches is that individual adjacency is maintained through *internal* lattice reconfiguration while it is lost in the *external* approach because it induces a migration mechanism. This makes it difficult to distinguish if performance changes are due to lattice reconfiguration or due to the induced migration.

Diversity measures are calculated differently. Phenotypic diversity is measured through their entropy, as opposed to considering fitness scores as a measure for diversity. Genotypic diversity is calculated using the Hamming distance.

Conditions needed to perform lattice reconfiguration are also different. *External* reconfiguration evaluates an evolution speed criterion that does not cover all possible scenarios for diversity at the phenotypic and genotypic spaces. It also requires an extra parameter to decide whether or not reconfiguration should occur. In contrast, *internal* lattice reconfiguration considers a wider range of generations for diversity measurements and performs reconfiguration if the difference between intervals is significant.

Two local selection methods were implemented, binary tournament presenting high selection pressure compared to anisotropic selection with low selection intensity. Through constant *internal* reconfiguration, binary tournament presents reduced flexibility to control selection intensity compared to anisotropic selection.

In general for the problems tackled here, *internal* lattice reconfiguration outperformed static and pre-programmed approaches in terms of convergence time for both local selection methods. However, in some cases, better hit rates were achieved through static and pre-programmed techniques. Statistical difference in convergence time was obtained for most problems with anisotropic selection and for half of them through binary tournament.

Adaptive approaches through binary tournament did not show significant improvement over constant *internal* reconfiguration in terms of efficiency and efficacy. Moreover, multiple comparison tests only show statistical difference in some of the problems. Similar results were obtained for anisotropic selection through adaptive approaches. However, with the exception of the SLE problem, the rest of the problems showed statistical proof in terms of convergence time. Also, adaptive *internal* reconfiguration achieved better performance in most of the problems.

Diversity tuning in cGAs for assessed benchmark problems was accomplished through dynamically changing their structural configuration, improving in many cases their overall performance. However, the effect of controlling the exploitation-exploration trade-off through cGAs structural properties was limited, as results thus far obtained have shown. Moreover, it was expected to obtain better results through adaptive approaches; thus the contribution to balancing the exploration-exploitation trade-off using diversity measures should be reconsidered due to their computational cost. Experimentally, the same or better performance was achieved through constant lattice reconfiguration for the problems tested here. This suggests that maintaining diversity through cGAs structural configuration does not need to be as prompt as phenotypic or genotypic changes occur. Constant modification of selection intensity during evolution will supply, at a certain level, an improved balance for diversity.

4.2 Dynamic cGAs Based on Local Selection

The aim in this section is to compare the dynamic control of the exploration-exploitation trade-off in cellular Genetic Algorithms (cGAs) from two perspectives: 1) through the approach proposed in previous section which is the dynamic reconfiguration of the population topology and thus to take advantage of their inherent structural properties; 2) through the local selection using a recently developed criterion known as anisotropic selection which allows to modify at local level the overall population selection pressure. In both perspectives, the dynamic control of selection pressure is implemented constantly (every c generations) or adaptively based on diversity changes at the phenotypic or genotypic spaces. Most benchmark problems in Appendix A are tackled and the statistical analysis is carried out to assess the convergence time.

This research has focused on cellular GAs and has aimed to provide a deeper insight into the flexibility they offer. Particularly, the structural properties of cGAs have been investigated in the previous and in the current chapter. The performance results of the proposed approaches have been empirically tested and statistically supported. In this final section, two perspectives for diversity tuning are studied and compared. Both are dynamic approaches that aim at controlling the overall selection pressure while taking advantage of the inherent structural properties of cGAs. Firstly, a dynamic approach to reconfigure the population topology is employed; internal cellular sub-structures presenting different NG ratios are implemented while the adjacency of individuals is maintained through their positions in the internal sub-structures [31, 33]. Secondly, diversity is maintained through the dynamic control of selection pressure induced by the local selection method. This is achieved using a local selection technique known as anisotropic selection [84]. In the next subsection both approaches are explained.

4.2.1 Selection Pressure Control

Two criteria are evaluated in order to dynamically control the selection pressure in cGAs. Firstly, through the population topology configuration, using the NGR as a tuning parameter, which is given by the local neighbourhood and the population size and shape. Regular local selection methods are applied during evolution. Secondly, through the local selection method, which can be tuned and modified in order to maintain an adequate exploration-exploitation trade-off.

4.2.1.1 Lattice Reconfiguration

In this section, the results obtained in the previous section as regards the dynamic reconfiguration of the grid are compared to controlling the induced selection pressure through a dynamic local selection method. In Subsection 4.1.3 the process for lattice reconfiguration occurs first between two different topology configurations that in average induce different levels of selective pressure. Then, a third topology is introduced $\left(2\left(\frac{n}{2} \times n\right)\right)$. Finally, a fourth lattice configuration is considered, with an average more exploitative behaviour $\left(n\left(1 \times \frac{n}{2}\right)\right)$. Both topologies are executed in horizontal and vertical alignments, with probability of occurrence of 50%. See Figure 4.1 for details.

In Subsection 4.1.4, the proposal of *internally* reconfigure the population topology in order to provide different levels of selection intensity is compared to Dorronsoro et al. approach for *external* lattice reconfiguration. Cases such as static, preprogrammed, constant and adaptive reconfiguration approaches are evaluated. The reader is referred to Section 4.1 for details on the lattice reconfiguration criteria that are compared in this section.

4.2.1.2 Dynamic Local Selection

Anisotropic selection depends on the direction of individuals location in a Von Neumann like local neighbourhood, which consists of individuals at the North, East, South and West directions, at a Manhattan distance of one from a central individual. Anisotropic selection would choose an individual following two probabilistic equations determined by an α parameter: $P_N = P_S = P_0\,(1+\alpha)$ and $P_E = P_W = P_0\,(1-\alpha)$ respectively. For example, if $\alpha = 0.0$ and $P_0 = 0.25$ a uniform probability for all neighbours is applied. Varying α from 0.0 to 0.9 assigns higher or lower probabilities to individuals located at North/South or East/West positions.

Similar to the previous approach, in Figure 4.3, the take-over times for anisotropic selection are drawn considering $\alpha \in [0.0, 0.9]$. These curves are obtained from an average of 100 experiments on a population of 400 individuals and a minimum α step of 0.1. Higher α values promote exploration while lower α values perform a more exploitative search. Moreover, for $\alpha \leq 0.3$ take-over times are very similar. Thus, for these values the expected effect of controlling the selection pressure is minimal. In the next subsection, configuration details for the proposed approaches are presented.

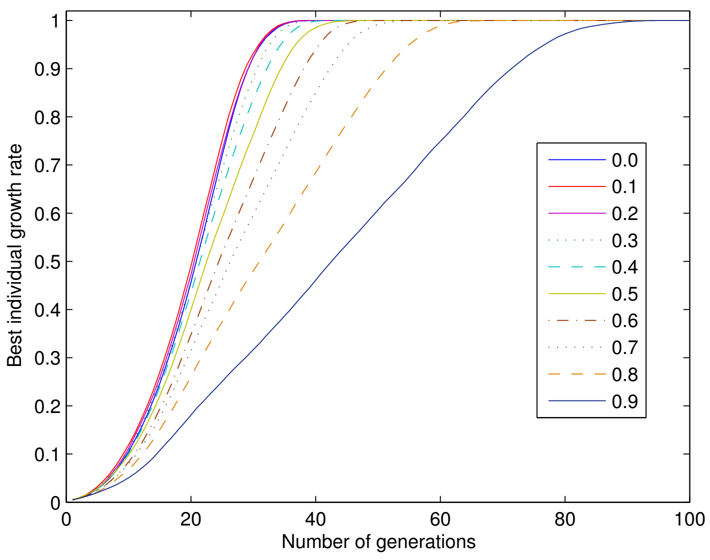


Figure 4.3: Take-over time for dynamic anisotropic local selection

4.2.2 Cellular Configuration

Two perspectives to dynamically control the selection pressure in cellular GAs are compared: 1) to reconfigure the population topology and inherently control the overall induced selection pressure; or 2) by local selection, a method known as anisotropic selection that allows an α parameter to be tuned and thus affect the exploration-exploitation trade-off. For the first perspective, results from Subsection 4.1.4 as regards dynamic *internal* lattice reconfiguration are analysed and compared to the second approach in this section.

For both approaches, the basic algorithmic structure is the same. The difference becomes clear when decision is made with respect to 1) reconfiguring the topology of the population or 2) increasing/decreasing the anisotropic parameter (α) in order to change the selection intensity making it more or less exploitative or explorative. These approaches are evaluated following two criteria: 1) the change is constant, which means that every c generations the lattice will be reconfigured or the α parameter is changed from 0.0 to 0.9 in a cyclic manner. 2) measures of diversity at the phenotypic or genotypic spaces are carried out every generation, and thus the lattice is reconfigured on a variable basis. Similar considerations apply to the α parameter in anisotropic selection. Measuring diversity changes at the phenotypic or genotypic spaces, a lower or higher α value is set in order to perform a more exploitative or more explorative search.

In Algorithm 12, the cGA pseudocode is described. A single random seed is used to generate the initial population. Initially, the population is evolved on a $4\left(\frac{2}{n} \times \frac{2}{n}\right)$ or on a $n \times n$ topology configuration or with anisotropic selection with $\alpha = 0.0$ (line 2). For each individual, neighbours located at North, East, South and West positions are evaluated in order to select a second parent for reproduction (line 7). For dynamic lattice reconfiguration, two local selection methods, presenting opposite selection pressures, have been applied. Binary tournament selection presents higher selection pressure when compared to constant anisotropic selection with $\alpha = 0.8$. For dynamic local selection, anisotropic selection with α variable is used. After synchronous updating (line 16), the population's average fitness score is verified as the stop condition (line 17), if the problem specific threshold has not been achieved, either lattice reconfiguration or dynamic anisotropic selection are performed either constantly or adaptively based on phenotypic or genotypic diversity measures (line 20 or line 21 respectively).

In Algorithm 13, constant lattice reconfiguration (CLR) and constant dynamic anisotropic (CDA) criteria are detailed. A constant interval of $c = 5$ generations is applied, to dynamically change the selection intensity. The same lattice configuration options drawn in Figure 4.1 are implemented and evaluated here. Rectangular $(2\left(\frac{n}{2} \times n\right))$ and linear $(n\left(1 \times \frac{n}{2}\right))$ lattice options are also executed in horizontal and vertical alignments with probability occurrence of 50%. For dynamic anisotropic selection, an initial $\alpha = 0.0$ is increased constantly by an $\alpha = 0.1$ step; once the limit is reached, α is reinitialized to 0.0.

Dynamic selection pressure based on phenotypic diversity applies the measure of the first experimental set-up presented in Subsection 4.1.2.2 which corresponds to the average population fitness score, $\Delta\bar{f}$, in consecutive generations. The same conditions are applied to the dynamic modification of the local selection through the α parameter of anisotropic selection. It is considered that too much exploitation would represent large changes on phenotypic values and that most individuals would be closely distributed over the landscape. On the contrary, excessive exploration would lead individuals to distant and possibly poor regions of the search space, and phenotypic values would present, in average, minimal changes. Both states are not desirable, instead a balance between exploration and exploitation should be achieved. Yet, as explained in Subsection 4.1.2.2 this criterion does not comprise all possible scenarios and therefore the

Algorithm 12 Dynamic lattice reconfiguration or anisotropic selection cGA

```
 1: procedure cGA
 2:     $n \times n \mid 4(\frac{n}{2} \times \frac{n}{2}) \mid \alpha = 0.0$    ▷ Initial topology or anisotropic selection config.
 3:     for $k \leftarrow 1, gens$ do
 4:         for $i \leftarrow 1, m$ do
 5:             for $j \leftarrow 1, n$ do
 6:                 $x_0 = x(i,j); f_0 = f(i,j);$
 7:                 $(f_n, f_e, f_s, f_w) \leftarrow evaluation(x_0, x_n, x_e, x_s, x_w);$
 8:                 $(x') \leftarrow selection(x_n, x_e, x_s, x_w);$
 9:                 $(x_1, x_2) \leftarrow recombination(x_0, x');$
10:                 $(x'_1, x'_2) \leftarrow mutation(x_1, x_2);$
11:                 $(f'_1, f'_2) \leftarrow evaluation(x'_1, x'_2);$
12:                 $(f'_0) \leftarrow [max|min](f_0, f'_1, f'_2);$
13:                 $(x_{temp}(i,j), f_{temp}(i,j)) \leftarrow replace(x'_0, f'_0);$    ▷ Replacement policy
    *if-better*
14:             end for
15:         end for
16:         $x = x_{temp}, f = f_{temp};$    ▷ Synchronous updating
17:         if $\bar{f} <= threshold$ then
18:             stop
19:         else
20:             $CLR|PLR(f)|GLR(x);$    ▷ Execute lattice reconfiguration mechanism
                                                                    OR
21:             $CDA \mid PDA(f) \mid GDA(x);$    ▷ Apply dynamic anisotropic selection
22:         end if
23:     end for
24: end procedure
```

Algorithm 13 Constant reconfiguration criterion

```
1: procedure CLR,CDA
2:     if $k\ mod\ c\ ==\ 0$ then                                   ▷ c is a constant
3:         if $[NGR = NGR_{n\times n} \mid \alpha = 0.9]$ then              ▷ Exploration limit
4:             $\left[NGR = NGR_{4(\frac{n}{2}\times\frac{n}{2})}; \mid \alpha = 0.0;\right]$        ▷ Restart constant cycle
5:         else
6:             $NGR = \left[\left(NGR_{2(\frac{n}{2}\times n)} \mid NGR_{n(1\times\frac{n}{2})}\right) \mid \alpha = \alpha + 0.1\right]$; ▷ Change $NGR$ OR increase $\alpha$
7:         end if
8:     else
9:     end if
10: end procedure
```

phenotypic entropy of the population is a desirable measure to determine diversity in the phenotypic space.

In Algorithm 14, phenotypic lattice reconfiguration (PLR) and phenotypic dynamic anisotropic (PDA) criteria are shown. If the difference between $\Delta\bar{f}_t$ and $\Delta\bar{f}_{t-1}$ is less than $\left(\Delta\bar{f}_{t-1} - \Delta\bar{f}_{t-2}\right)$, individuals are too spread and exploitation should be promoted by reconfiguring the population topology or decreasing the α value. If the same condition indicates an increase in the average fitness score, exploration should be promoted through a lattice configuration that presents lower selective pressure or higher α value. Otherwise, the population topology remains the same.

Algorithm 14 Phenotypic diversity criterion

```
1: procedure PLR/PDA($f$)
2:     if $\Delta\bar{f}_t < (2 * \Delta\bar{f}_{t-1}) - \Delta\bar{f}_{t-2}$ then
3:         $[NGR = NGR_{higher} \mid \alpha = \alpha - 0.1]$                          ▷ Exploit
4:     else
5:         if $\Delta\bar{f}_t > (2 * \Delta\bar{f}_{t-1}) - \Delta\bar{f}_{t-2}$ then
6:             $[NGR = NGR_{lower} \mid \alpha = \alpha + 0.1]$                      ▷ Explore
7:         else
8:         end if
9:     end if
10: end procedure
```

The same form to determine genotypic diversity used in Subsection 4.1.2.3 is applied here. The Hamming distance among chromosomes is used as a measure. Thus, if difference among genotypes at current ($\Delta\bar{H}_t$) and previous ($\Delta\bar{H}_{t-1}$, $\Delta\bar{H}_{t-2}$) generations is significant, the population topology is reconfigured. In Algorithm 15 details for genotypic lattice reconfiguration (GLR) and genotypic dynamic anisotropic (GDA) criteria are presented: if $\left(\Delta\bar{H}_t - \Delta\bar{H}_{t-1}\right)$ is less than $\left(\Delta\bar{H}_{t-1} - \Delta\bar{H}_{t-2}\right)$, diversity is being lost, therefore exploration should be promoted through the reconfiguration of the grid by reducing the NGR or increasing the α value. On the contrary, if $\left(\Delta\bar{H}_t - \Delta\bar{H}_{t-1}\right)$ is grater than $\left(\Delta\bar{H}_{t-1} - \Delta\bar{H}_{t-2}\right)$, diversity is increased and exploitation should be encouraged. Therefore, the grid configuration is changed to another with higher ratio or smaller α value. If neither of these conditions are satisfied, the population topology or α value remain the same.

Algorithm 15 Genotypic diversity criterion

```
1: procedure GLR/GDA($x$)
2:     if $\Delta\bar{H}_t > (2 * \Delta\bar{H}_{t-1}) - \Delta\bar{H}_{t-2}$ then
3:         $[NGR = NGR_{higher} | \alpha = \alpha - 0.1]$          ▷ Exploit
4:     else
5:         if $\Delta\bar{H}_t < (2 * \Delta\bar{H}_{t-1}) - \Delta\bar{H}_{t-2}$ then
6:             $[NGR = NGR_{lower} | \alpha = \alpha + 0.1]$          ▷ Explore
7:         else
8:         end if
9:     end if
10: end procedure
```

4.2.3 Results Analysis

In this subsection the experimental results are analysed. Most benchmark problems included in Appendix A are evaluated and compared here. The same experimental constraints, used in previous sections, are also implemented; as well as the same configuration for genetic operators is used: Single Point Crossover (SPC) with probability $P_c = 1.0$ and mutation with probability $P_m = 0.02$.

The following experimental constraints are evaluated:

- A population size of 400 individuals is used for most problems, except for the

GPS and the MTTP problems, where a population size of 64 and 100 individuals are used respectively.

- Local neighbourhood configuration is Von Neumann composed by four individuals plus the central one.
- 100 independent runs are carried out for each proposed criterion.
- A limit of 500 generations is defined for most problems, except for the Langerman function, the SLE and the MMDP problems where the limit is set to 700 generations.

The statistical analysis initially performs a normality test in each set of experimental results as regards the convergence time. Thus, normality is determined by the Kolmorov-Smirnov test or the Lilliefors test, both with 5% of significance. The Lilliefors test is suitable when a fully-specified null distribution is unknown, contrary to the Kolmorov-Smirnov test. Once the normality of the results has been established, the Analysis of Variance (ANOVA) is applied to results with normal distribution whereas the Kruskal-Wallis test is applied in any other case.

As a reference for the analysis of results, the performance of a static square topology is used. Thus, statistical significance tests evaluate the performance between the proposed dynamic criteria and a standard cGA. To represent that statistical significant difference at 5% has been determined in terms of convergence time, a + symbol is used. Instead, a • symbol represents results are not statistically different, and therefore applying the proposed dynamic criteria makes no difference in terms of the number of generations, when compared to a standard cGA. On the other hand, having statistically different results does not mean the proposed approaches improve the performance of a standard cGA but highlights that results are indeed different. Therefore to avoid confusion an individual analysis for each problem accompanies the interpretation of the results.

Results are analysed based on convergence time and hit rate. The convergence time is measured in terms of the number of generations while the hit rate represents the number of experiments that succeeded in solving the problem, out of the total number of experiments which is set to one hundred. On each table of results, bold fonts highlight the best performance in terms of efficiency and efficacy. Results accuracy is assured by the stop condition which is for all problems highly accurate, see Appendix A for details. Commonly, as a stop condition, researchers use a maximum limit in the number of generations or when an individual has reached the global optimum. However, in this research the stop condition evaluates that successful experimental samples should have achieved a solution of the same quality. The stop condition evaluates the average population fitness score with respect to a problem-specific threshold. This condition has been defined due to previous research constraints related to the fault tolerant arena. Although that is a different topic, decision was made in order to maintain the same stop condition, to be consistent with previous research.

4.2.3.1 Continuous problems

The results for continuous problems tackled through binary tournament selection, constant anisotropic selection ($\alpha = 0.8$) and dynamic anisotropic selection are presented in Tables 4.13, 4.14 and 4.15 respectively. The effect of being highly exploitative (binary tournament) or highly explorative (constant anisotropic selection) are easily observed in terms of convergence time through both static local selection methods while using a static square topology.

For the Rastrigin function, which is highly multi-modal, 100% hit rate is achieved in all cases and the best convergence time is obtained through binary tournament selection. However, statistical significance is proved through lattice reconfiguration for constant and dynamic anisotropic selection. The best convergence time is achieved by constant lattice reconfiguration. On the other hand, for the Griewank function, which is highly multi-modal and epistatic; applying binary tournament selection affects negatively the hit rate when the lattice is reconfigured. In contrast, for constant anisotropic selection ($\alpha = 0.8$), both performance measures are enhanced in all cases. The hit rate improves from 67% to 95% through genotypic based reconfiguration; while convergence time also improves with statistical significance. Similar results are obtained through dynamic anisotropic selection presenting overall the best hit rate of 98% achieved when measuring the phenotypic diversity.

The last continuous problem, the Langerman function, which adds the difficulty of non-regularity, shows in general a very low hit rate. Slight improvements in terms of convergence time and hit rate are obtained when lattice reconfiguration or dynamic local selection are applied. Yet, results statistical significant difference is shown. The hit rate improves from 21%, on a static square topology, to 41% obtained through

Table 4.13: Convergence time[1] and hit rate[2] for continuous problems through constant and dynamic internal lattice reconfiguration and binary tournament local selection

Dynamic control	Rastrigin	Griewank	Langerman
None	136.69 ± 12.42 100%	275.65 ± 44.56 **90%**	291.04 ± 103.90 21%
Constant	$\mathbf{124.08} \pm 13.16$ 100%	$\mathbf{257.23} \pm 57.03$ 68%	$\mathbf{222.90} \pm 109.72$ 20%
Phenotypic	129.13 ± 14.62 100%	270.54 ± 55.15 77%	230.96 ± 96.96 **25%**
Genotypic	126.19 ± 10.09 100%	264.16 ± 49.22 82%	248.81 ± 118.69 22%
ANOVA/K-W [3]	•	(+)	•

Table 4.14: Convergence time[1] and hit rate[2] for continuous problems through constant and dynamic internal lattice reconfiguration and anisotropic local selection

Dynamic control	Rastrigin	Griewank	Langerman
None	230.38 ± 15.51 100%	444.73 ± 41.52 67%	432.61 ± 47.01 21%
Constant	$\mathbf{188.91} \pm 15.96$ 100%	402.91 ± 47.46 93%	$\mathbf{309.42} \pm 81.44$ 16%
Phenotypic	196.71 ± 29.43 100%	$\mathbf{400.56} \pm 45.63$ 91%	364.33 ± 60.92 **33%**
Genotypic	202.92 ± 16.62 100%	396.14 ± 46.87 **95%**	356.46 ± 58.23 26%
ANOVA/K-W [3]	(+)	(+)	(+)

Table 4.15: Convergence time[1] and hit rate[2] for continuous problems through dynamic anisotropic local selection

Dynamic control	Rastrigin	Griewank	Langerman
Constant	$\mathbf{194.15} \pm 13.65$ 98%	398.59 ± 42.02 97%	346.31 ± 82.05 22%
Phenotypic	237.50 ± 30.02 100%	$\mathbf{397.19} \pm 41.64$ **98%**	362.70 ± 69.06 **41%**
Genotypic	213.19 ± 27.79 100%	422.96 ± 52.15 84%	$\mathbf{259.30} \pm 83.93$ 26%
ANOVA/K-W [3]	(+)	(+)	(+)

1 Convergence time is the average number of generations for successful experiments ± std. dev.

2 Hit rate is presented as the percentage of successful experiments out of hundred samples.

3 Statistical tests: (+) statistical difference is proved / • is not proved.

dynamic anisotropic selection approach based on phenotypic diversity.

4.2.3.2 Real and combinatorial problems

In Tables 4.16, 4.17 and 4.18 results for real and combinatorial problems are presented. Similar to results obtained when tackling continuous problems, the difference between applying binary tournament or anisotropic ($\alpha = 0.8$) local selection in terms of the average convergence time is clear in most of the problems except for the GPS attitude determination problem.

The FMS problem presents similar hit rates for constant local selection methods. For binary tournament, the improvement is minimal while a significant increase in hit rate (from 46% to 71%) is obtained through anisotropic selection with constant $\alpha = 0.8$ when the population topology is constantly rearranged. The convergence time is also reduced with significant difference while standard deviation shows the results consistency. There is also an improvement in terms of hit rate through dynamic anisotropic selection when compared to the lattice reconfiguration with binary tournament and constant anisotropic selection. Comparing constant and dynamic anisotropic selection based on phenotypic diversity both present a hit rate of 61%. However, a better convergence time is obtained through dynamic lattice reconfiguration with constant anisotropic selection.

Solving a SLE of ten variables is a difficult task; low hit rates, below 50%, are obtained in all cases. Higher selection pressure, binary tournament, performs better than constant anisotropic selection on a static square topology. The results are the opposite for dynamic lattice reconfiguration mechanisms with binary tournament and constant anisotropic selection. Higher selection pressure negatively affects the hit rate meanwhile lower selection intensity improves the hit rate from 19% to 38%, being more consistent to constantly rearrange the population topology. For dynamic anisotropic selection, the best overall hit rate (44%) obtained through phenotypic diversity is achieved with no statistically significant improvement in terms of convergence time.

For the GPS attitude determination problem, there is no statistical significance in convergence time between having a static squared topology and the proposed dynamic lattice reconfiguration criteria. However, the best hit rate (87%) is obtained through anisotropic local selection and phenotypic diversity based criterion. Yet, this result is similar to applying dynamic anisotropic local selection (84%).

Finally, for combinatorial problems, results show the following: the MMDP presents no improvement in terms of hit rate when lattice reconfiguration or dynamic anisotropic

Table 4.16: Convergence time[1] and hit rate[2] for real and combinatorial problems through constant and dynamic lattice reconfiguration with binary tournament local selection

Dynamic control	FMS	SLE	GPS	MMDP	MTTP
None	210.78 ± 58.76	297.28 ± 72.56	73.12 ± 41.83	431.31 ± 54.44	259.65 ± 44.16
	41%	**39%**	57%	89%	99%
Constant	189.78 ± 66.99	253.05 ± 60.38	**64.66** ± 25.78	**361.71** ± 65.24	252.16 ± 65.18
	42%	19%	50%	87%	100%
Phenotypic	192.58 ± 56.96	**250.09** ± 55.55	80.55 ± 54.25	384.38 ± 72.38	248.46 ± 56.91
	41%	21%	**60%**	89%	100%
Genotypic	**175.00** ± 61.98	274.88 ± 98.10	65.24 ± 34.22	388.95 ± 62.14	**239.17** ± 38.23
	45%	27%	57%	**90%**	100%
ANOVA/K-W [3]	●	(+)	●	(+)	●

1 Convergence time is measured as the average number of generations for successful experiments. Corresponding standard deviation is included after ± symbol.

2 Hit rate is presented as the percentage of successful experiments out of the total number of experiments which is one hundred samples.

3 Statistical tests: (+) statistical difference is proved / ● is not proved.

Table 4.17: Convergence time[1] and hit rate[2] for real and combinatorial problems through constant and dynamic lattice reconfiguration with constant anisotropic local selection ($\alpha = 0.8$)

Dynamic control	FMS	SLE	GPS	MMDP	MTTP
None	379.38 ± 67.83	594.21 ± 54.23	**80.73** ± 20.36	568.98 ± 54.01	444.45 ± 36.92
	46%	19%	76%	**97%**	31%
Constant	**300.32** ± 53.09	506.23 ± 22.29	86.76 ± 25.15	**558.56** ± 57.01	**403.00** ± 47.85
	71%	**38%**	80%	91%	**87%**
Phenotypic	321.91 ± 67.11	526.63 ± 84.10	81.75 ± 23.07	608.13 ± 47.26	428.17 ± 50.21
	61%	**38%**	**87%**	87%	75%
Genotypic	328.76 ± 67.82	**489.84** ± 100.97	80.00 ± 19.15	615.22 ± 52.97	425.02 ± 47.48
	56%	26%	77%	85%	70%
ANOVA/K-W [3]	(+)	(+)	●	(+)	(+)

1 Convergence time is measured as the average number of generations for successful experiments. Corresponding standard deviation is included after ± symbol.

2 Hit rate is presented as the percentage of successful experiments out of the total number of experiments which is one hundred samples.

3 Statistical tests: (+) statistical difference is proved / ● is not proved.

Table 4.18: Convergence time[1] and hit rate[2] for real and combinatorial problems through dynamic anisotropic local selection

Dynamic control	FMS	SLE	GPS	MMDP	MTTP
Constant	**324.21** $\pm$ 65.24	**511.73** $\pm$ 69.36	**68.93** $\pm$ 25.20	**590.08** $\pm$ 49.01	**394.42** $\pm$ 48.42
	65%	38%	76%	**92%**	**95%**
Phenotypic	363.06 $\pm$ 63.17	515.70 $\pm$ 87.26	85.85 $\pm$ 33.83	641.14 $\pm$ 34.43	419.92 $\pm$ 46.49
	61%	**44%**	**84%**	50%	66%
Genotypic	327.29 $\pm$ 66.67	530.88 $\pm$ 96.49	74.51 $\pm$ 30.75	643.13 $\pm$ 39.60	440.00 $\pm$ 35.62
	64%	27%	82%	38%	51%
ANOVA/K-W [3]	(+)	●	(+)	(+)	(+)

1 Convergence time is measured as the average number of generations for successful experiments. Corresponding standard deviation is included after $\pm$ symbol.

2 Hit rate is presented as the percentage of successful experiments out of the total number of experiments which is one hundred samples.

3 Statistical tests: (+) statistical difference is proved / ● is not proved.

selection are applied. For the second criterion, the hit rate drops dramatically to 38% for genotypic diversity based dynamic selection. There is a statistically significant improvement in terms of convergence time for binary tournament selection when the lattice reconfiguration is carried out based on phenotypic or genotypic diversity. Similar results are obtained for the MTTP; overall binary tournament provides the highest hit rate with no improvement in convergence time. However, constant anisotropic selection on a standard cGA topology shows the worst hit rate (31%) which significantly improves to 87% and 95% through constant lattice reconfiguration and constant dynamic anisotropic selection respectively.

4.2.4 Summary

In this section, two approaches for diversity tuning in cellular GAs have been evaluated and compared. The aim was to dynamically control the overall induced selection pressure while taking advantage of the structural properties of cGAs.

Two constant local selection methods were applied while dynamic lattice reconfiguration was performed: binary tournament and anisotropic selection. Both methods present distinctively opposite selection pressure. The effect of dynamically reconfigure the grid whereas binary tournament local selection was applied was not significant in terms of efficiency. Binary tournament selection is highly exploitative while constant anisotropic selection becomes more explorative. In fact, anisotropic selection offers more flexibility when internal lattice reconfiguration is performed. Therefore, anisotropic selection with constant $\alpha = 0.8$ showed better convergence times in most of the problems except for the MMDP.

Implementing dynamic anisotropic selection provided similar or better overall performance in comparison to reconfiguring the population topology. Dynamic anisotropic selection showed greater flexibility in terms of selection pressure. Therefore, highly explorative search could be performed at the cost of increasing the convergence time. However, for some problems, having high and constant selection pressure produced a better performance (the Griewank function and the SLE problem). But, in most problems, if selection pressure was dynamically controlled the overall performance was enhanced.

4.3 Chapter Summary

This chapter was aimed at investigating the dynamic reconfiguration of the population topology and the dynamic allocation of probabilities in anisotropic selection as mechanisms for diversity tuning to provide a more balanced exploration-exploitation trade-off. Three experimental set-ups were assessed in order to validate the proposed criteria. The contribution to knowledge in this arena is:

- Dynamic internal lattice reconfiguration to allow different levels of selective pressure while maintaining the adjacency of individuals; any improvement in the algorithmic performance is due to the reconfiguration mechanism without involvement of other evolutionary based operations.

- Improvement in cGAs performance through dynamically reconfigure the population topology was achieved for most benchmark problems. Two local selection methods presenting opposite selective pressures were locally applied. In most benchmark problems (except the Griewank function and the MMDP problem) the average number of generations required to find the solution was reduced and in most cases the hit rate increased. Both performance measures were statistically assessed.

- Having a number of levels for selective pressure through different topology configurations result in a more effective attempt for balancing the exploration-exploitation trade-off from a structural point-of-view.

- A comparative analysis between: 1) Dorronsoro et al. proposed mechanism for externally changing the population topology [6], and 2) the internal lattice reconfiguration criterion proposed in this research; showed that the former achieved better efficiency in most benchmark problems with binary tournament selection (high selective pressure) through adaptive measures. However, Dorronsoro et al. mechanism implies individuals migration which benefits local diversity and implicitly modifies the selective pressure. On the other hand, with anisotropic selection (low selective pressure with $\alpha = 0.8$), cGAs performance is improved in terms of efficiency and efficacy except for the MMDP problem; and constant lattice reconfiguration outperformed adaptive measures.

- In cGAs exploration is mainly carried out globally throughout the grid and exploitation is mostly promoted locally within neighbourhoods. Therefore, the proposed mechanism for diversity tuning works at a global level through the configuration of the lattice and its effect does not need to be a prompt response to phenotypic or genotypic diversity changes. Constant and periodical reconfiguration of the grid provides a similar effect as measuring diversity in any of both spaces to trigger the reconfiguration mechanism and is less computationally expensive.

- Anisotropic selection assigns probabilities for selection according to individuals locations in the neighbourhood, where exploitation is mainly promoted in cGAs. Dynamically modifying those probabilities for selection of individuals in the neighbourhood induces different levels of selective pressure. An improvement in cGAs performance is achieved in terms of efficiency and efficacy for most benchmark problems except for the MMDP problem.

- Applying dynamic lattice reconfiguration and dynamic anisotropic selection achieved similar algorithmic performance in terms of efficiency in most benchmark problems. Lattice reconfiguration affects globally the exploration of the search space carried out throughout the grid while dynamic anisotropic selection affects the exploitation promoted locally within neighbourhoods. Therefore a combination of both approaches is desirable and will be explored in future work.

Chapter 5

On 3D Cellular Genetic Algorithms

This chapter aims to analyse and compare 2D and 3D cellular GAs, while maintaining in general their configuration constraints such as population size, neighbourhood radius, local selection method, replacement polices, among others. A primary objective is to provide a wide insight into the advantages of increasing cellular dimensionality for future development of 3D adaptive optimization engine architectures.

Parallel architectures have been suitable to adapt and implement evolutionary search strategies. Particularly, fine or cellular Genetic Algorithms own a processing structure that creates a strong association between soft (algorithm) and hard (architecture) levels, with attractive properties in terms of system performance. Typically, a cellular GA consists of a decentralized population distributed on a grid structure with its edges wraparound, following a toroidal shape. Only one individual is placed at each grid position. Thus, interaction among individuals is performed at two stages; locally, each individual is mated with its closest neighbours, and globally through genotypes information that is spread all around the structure.

Cellular Genetic Algorithms present several advantages over other GAs approaches; such as the diversity level which can be maintained for much longer in comparison with centralized ones [3]. Moreover, cellular GAs outperform panmictic based GAs, not only in terms of efficiency, measured as the number of generations required to reach a defined threshold; but also in its efficacy, quantified as the hit rate in combination with the results accuracy. These improvements emerge from the specific interaction among individuals defined by the population mapping over a grid structure.

An immediate step towards a better understanding of cGAs behaviour and model their performance, is to increase their structural dimensionality; which is the main purpose of the study herein presented. Comparing 2D and 3D square shape topologies while keeping the same processing and interaction constraints among individuals will provide a wider overview of cGAs potential as optimization tools.

Previously in [28], a study on multi-dimensional cellular automata was carried out. The interaction among cells was performed using genetic operations such as selection, recombination and mutation. Three problems which aim is to evolve local rules to perform a specific global behaviour were tackled. Those problems are the majority problem, the checker board problem and the evolution of bitmaps. Different parameter settings in terms of crossover and mutation probabilities and the size of the tournament in local selection were evaluated. In most of the problems a 3D topology outperformed 1D and 2D topologies in terms of maximum fitness score achieved and number of generations. However, comparison among structural dimensions is carried out with very different number of cells, for example, the majority problem was assessed using a 13×13 2D topology and a $7 \times 7 \times 7$ 3D topology, which represents a difference of 274 cells.

In this chapter the aim is to further investigate the effect of having a 3D-cGA while using different population sizes and local neighbourhood distances or radius while tackling difficult landscapes and larger problem sizes. The population sizes used in this study are very similar when using 2D and 3D structures. An initial set of four problems with similar characteristics and distinct landscapes is assessed. Rastrigin and Schwefel functions are multi-modal and separable. On the other hand, Ackley and Griewank functions are not only multi-modal and regular but also epistatic. Details of these benchmark problems are provided in Appendix A. A reduced test bench is initially assessed from a dimensional point-of-view with problems of medium to high difficulty for GAs, such as those tackled here while using different population sizes. Research on dimensionality in cellular GAs presented in this thesis is the starting point for future investigation at the System Level Integration research group.

In addition, a major motivation for this research is to explore at algorithmic level the benefits of 3D structures in cellular GAs and to relate them with those of the recently developed three-dimensional integration technology. 3D integration technology research started in the early 90s. Recent advances in this area have presented positive results at hard level; hence combining the soft approach of implementing 3D cellular GAs as optimization engines, in order to solve hard real time problems, would bring together the advantages that 3D integration technology has provided. Although, it is not yet a widely commercial technology; 3D integration technology is considered as the future for coarse and fine grained reconfigurable architectures [90, 91]. Among the advantages that 3D integration technology brings are: reduction of the routing length, decrease of the interconnections delay which impact not only the fabric size but also the device performance. It has also been reported a significant improvement in terms of logic and memory density. Regarding logic density, for fine grained devices, it has been determined that 80%-90% of their area is used for reconfigurable interconnections. This percentage is reduced to 25%-60% when 3D integration technology is applied [92].

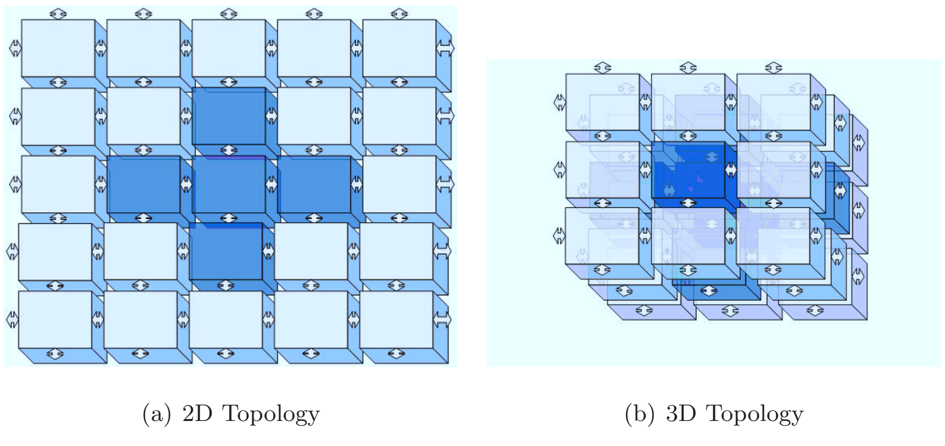
Future adaptive systems must offer characteristics such as fast adaptation, autonomous behaviour and fault tolerance. Cellular GAs have shown to be adaptive as well as fault tolerant for specific applications [93, 53]. 3D cellular architectures will offer the added advantage of speed and package density. This paper investigates a number of 3D cellular GA architectures and compares these to their 2D counterparts.

5.1 Algorithm configuration

Most of the existing cGAs studies are based on one and two dimensions. The goal of this study is to compare 3D versus 2D cellular GAs in terms of efficiency and efficacy. The population is structured as a square toroidal 2D grid in 2D-cGAs and as a cube toroidal 3D grid in 3D-cGAs, as shown in Figure 5.1.

Several population sizes are considered for both grids (2D and 3D). The population sizes for 2D grids are (3×3), (5×5), (8×8), (11×11), (15×15), and (19×19). While the population sizes introduced in 3D are $(2 \times 2 \times 2)$, $(3 \times 3 \times 3)$, $(4 \times 4 \times 4)$, $(5 \times 5 \times 5)$, $(6 \times 6 \times 6)$, and $(7 \times 7 \times 7)$. These sizes are selected to produce almost equal population sizes for both grids.

Chromosomes are encoded as binary strings. Each gene (variable) has a length of 10-bits, so the chromosome length L is equal to $10 \times q$ bits, where q is the function dimension. Although the considered neighbourhood topology is linear for both grids, the neighbourhood size differs according to the grid dimensions. Thus, the algorithm



(a) 2D Topology (b) 3D Topology

Figure 5.1: Cellular toroidal topologies

is configured with two different neighbourhood radii in both dimensions. Considering one distance step from the central cell results in 4 neighbours (North, East, West, and South) with radius 0.89 in case of 2D-cGA and 6 neighbours (horizontal North and South, vertical North and South, East, and West) with radius 0.925 in case of 3D-cGA.

Figure 5.2 shows the NGR considering two different radii and lattices. Smaller neighbourhood size yields lower NGR which in turn decreases as the population size increases. A lower NGR implies less global selection intensity and therefore more exploration [31, 82, 33]. As show in Figure 5.3 the NGR is not evaluated for a population size of (3×3) (2D) with three distance steps because of the small grid size. Proceeding three steps from an individual results in increasing its probability for selection. For the same reason, the NGR is not computed for population sizes of less than $(6 \times 6 \times 6)$ individuals.

Figure 5.2 shows the proportion of the best individual growth curves for population sizes (19×19) (2D) and $(7 \times 7 \times 7)$ (3D) considering one and three distance steps. As the curves imply, 2D-cGA with one step has the slowest growth curve while 3D-cGA with three steps has the fastest growth curve. 2D-cGA with three steps ($NGR = 0.2679$) produces almost similar growth curve to the 3D-cGA with one step ($NGR = 0.2673$). The reason of this behaviour is that in both cases the ratio is similar [31]. In the following subsection, the cellular GA pseudo-code evaluated in this chapter is presented.

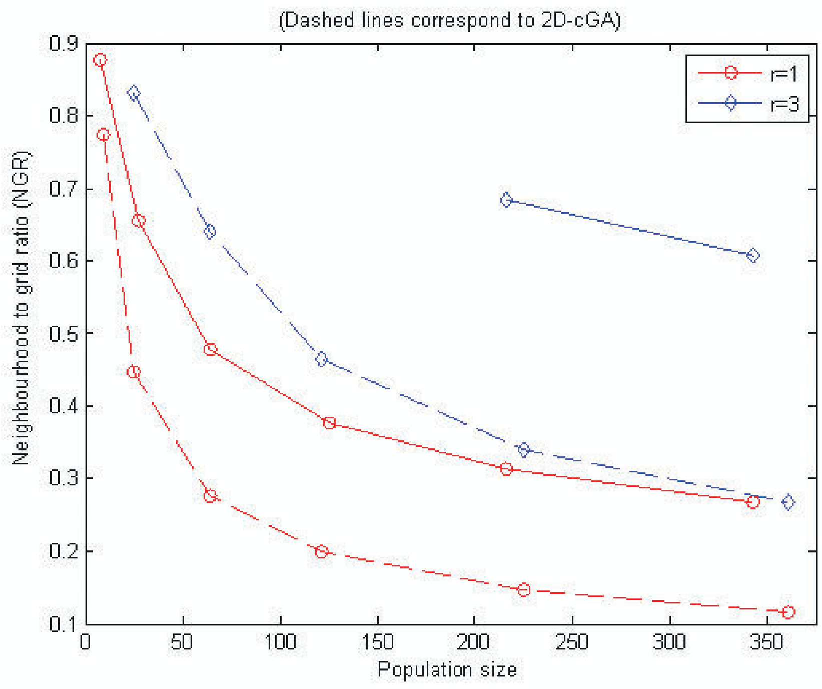


Figure 5.2: 2D/3D Neighbourhood to grid ratio (NGR) with different population sizes

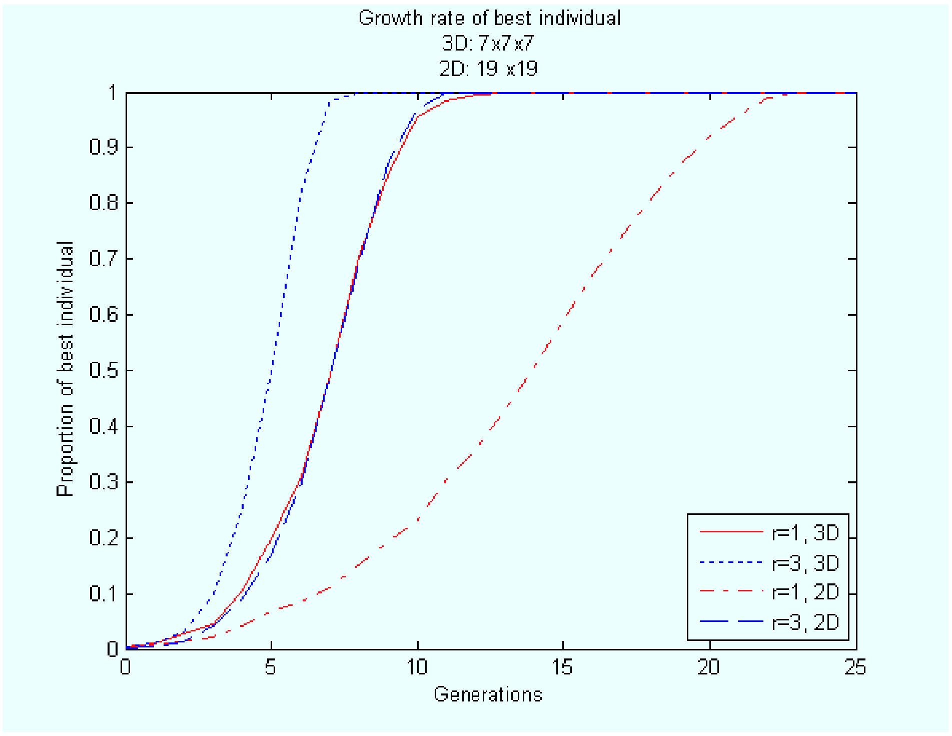


Figure 5.3: 2D/3D Best individuals growth curves

5.1.1 3D cellular GA Pseudo-code

In Chapter 2, Section 2.4, the basic pseudocode for a 2D-cGA is presented. In Algorithm 16, a similar pseudocode with variations corresponding to a 3D-cGA is shown. The initial stages are the same than in a 2D-cGA. A random initial population is generated, which then is evaluated and proceed by successively updating the individuals using genetic operations, until a termination condition is fulfilled. The main difference is the configuration of the local neighbourhood, which in 3D adds 2 individuals for a L7 neighbourhood.

The local selection method used in the neighbourhood is binary tournament (BT) to select the second parent (line 7) while the first parent is the current one (line 6). A single point crossover for recombination is applied with probability $P_c = 0.9$ -line 8- that delivers one child with the best fitness. Following a bit-flip mutation with probability $P_m = 0.02$ (line 9).

The new individual generated by local selection, crossover, and mutation replaces the old one if it is better and added into a temporal population (lines 11). Considering synchronous individuals updating, each population replaces the previous one completely (line 13).

In the next section experimental constraints and results analysis are provided.

5.2 Results Analysis

The next experimental constraints are defined:

- Local neighbourhood L5 with Manhattan distance of 1 and 3
- Local selection: 1st parent: central individual + 2nd parent: binary tournament
- Single Point Crossover with probability $P_c = 0.9$
- Mutation with probability $P_m = 0.02$
- Maximum number of generations: 500
- Chromosome length: 10 bits $\times$ problem dimension
- Replacement policy: replace if-better
- Stop condition: average fitness $\leq$ problem specific threshold

Algorithm 16 3D Cellular Genetic Algorithm Pseudocode

```
1:  procedure CGA
2:      $(x) \leftarrow random\,(x_0)$                                    ▷ initial population
3:      $(f) \leftarrow evaluation\,(x)$                                  ▷ evaluation
4:      while $k \leftarrow 1, generations,$  or  $\bar{f} <= threshold$ do
5:          for $i \leftarrow 1, populationSize$ do
6:              $(f_0, x_0) \leftarrow (f_i, x_i)$                        ▷ current individual
7:              $(f_2, x_2) \leftarrow selection\,(f, f_{front}, f_{back}, f_N, f_E, f_S, f_W)$   ▷ 2nd parent
    selection
8:              $(x'_1, x'_2) \leftarrow recombination\,(x_0, x_2)$       ▷ parents recombination
9:              $(x''_1, x''_2) \leftarrow mutation\,(x'_1, x'_2)$        ▷ offspring mutation
10:             $(f_{new}, x_{new}) \leftarrow evaluation\,(x_0, x''_1, x''_2)$ ▷ current individual and offspring
    evaluation
11:             $(f_{temp}, x_{temp}) \leftarrow replacement\,(f_{new}, x_{new})$   ▷ individuals replacement
12:         end for
13:         $(f, x) \leftarrow (f_{temp}, x_{temp})$          ▷ temporary population replaces current one
14:     end while
15: end procedure
```

The results are assessed following two metrics: the mean number of generations, algorithm efficiency, and the hit rate in combination with the results accuracy represent the efficacy of the algorithm. Both are computed as an average of one hundred independent runs.

The cGA is applied to the Rastrigin and Schwefel functions with dimension $n = 10$, and neighbourhood radii $(0.8944, 2.0755$ and $0.9258, 2.1026)$ for 2D and 3D respectively. The neighbourhood radii are almost similar for both topologies considering the same distance steps; the slight difference refers to a grid connection which assigns six neighbours in the 3D grid instead of four neighbours in the 2D grid considering one distance step [28].

In Figure 5.4 the average number of generations for the Rastrigin and Schwefel functions with a neighbourhood radius with Manhattan distance equal to three is shown. The results obtained are almost similar for both grid topologies (2D and 3D) with respect to the average number of generations considering both radii. An increase in the average number of generations is noticed with 3D-cGA considering the Rastrigin function with Manhattan distance of 3. The hit rates of previous configuration are shown in figure 5.5. Similar hit rates are obtained especially for larger population sizes.

In contrast, the results obtained from applying the same algorithmic configuration to the Griewank and the Ackley functions for $q = 5$ dimensions show an improvement for 3D-cGAs in convergence time but in terms of hit rate only for the Griewank function. In Figures 5.6 and 5.7 results for the Ackley function are presented; in terms of the average number of generations, 3DcGA improves 2DcGA performance with around 10 generations for small array sizes, and with around 20 for largest populations when a local neighbourhood radius $r = 1$ is applied. For $r = 3$ a similar profile is obtained; however in this case results are compared only for population sizes with more than 200 individuals, due to the fact that large local radii on 3D structures represent individual duplicity during selection and therefore are omitted for comparison.

Regarding the hit rate, 2D-cGAs outperform 3D-cGAs for populations over 100 individuals with $r = 1$. Similar results are observed with $r = 3$, although the difference is reduced to approximately 10%.

In Figure 5.8, results for the Griewank function are presented considering a $n = 5$ dimension. For this case, the 3D-cGA convergence time improvement is comparatively minor than with the Ackley function. The hit rate obtained represents an improvement

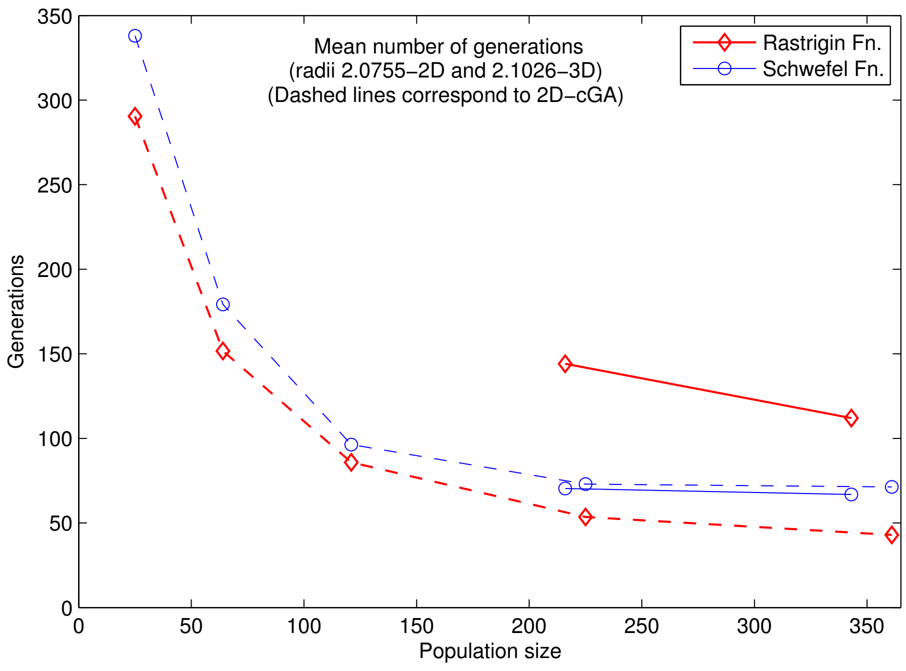


Figure 5.4: Rastrigin and Schwefel functions average number of generations

of approximately 40% for small population sizes, see Figure 5.9. However, when the population size increases to more than 100 individuals both cGA approaches perform similarly.

In conclusion, the 3D-cGA outperforms the 2D-cGA when solving the Griewank problem and achieves better convergence time for the Ackley function; while both perform similarly when solving the Rastrigin and the Schwefel problems. As mentioned previously, all these problems are highly multi-modal and regular but the Rastrigin and Schwefel functions are separable while the Griewank and Ackley are non-separable which make them more difficult to optimize. As explained before, with 3D lattices the neighbourhood size is more dense than in 2D, considering similar population sizes and distance steps. The larger the neighbourhood size the higher the selection intensity and therefore a more exploitative search. This implies finding the best solution faster (less number of generations) which explains the convergence time reduction by 3D-cGA.

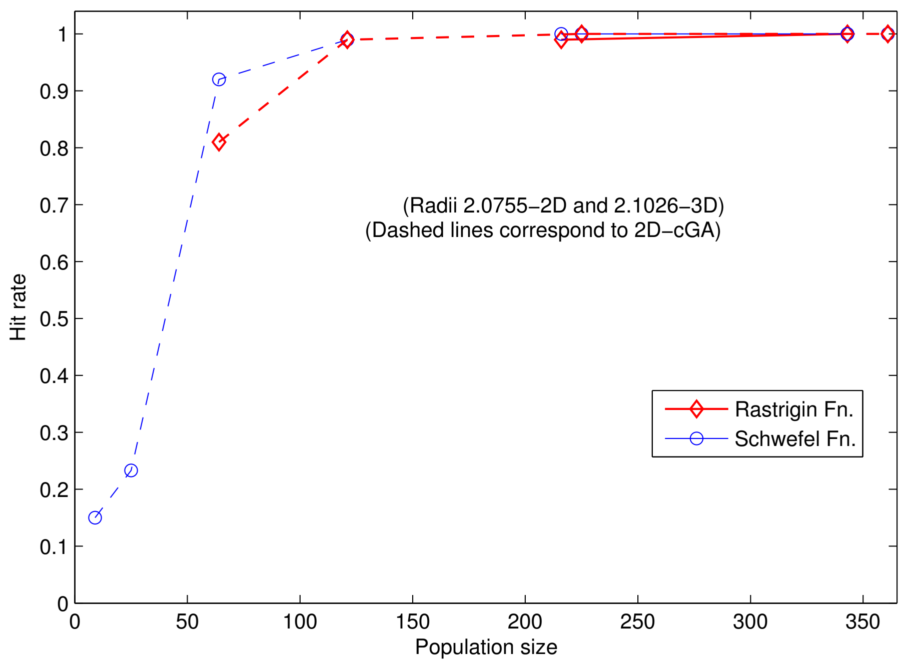


Figure 5.5: Rastrigin and Schwefel functions hit rate

5.3 Summary

This chapter has analysed and compared the performance of cellular GAs using 2D and 3D topologies while tackling a set of four benchmark problems presenting characteristics such as multi-modality and epistasis. The results obtained by 3D-cGAs empirically showed to be more efficient in terms of convergence time when solving harder problems, the Griewank or the Ackley functions, which are multi-modal and epistatic problems. In terms of the hit rate, both cellular structures achieve similar percentages but a 3D-cGA improves when smaller local neighbourhood radius is used. A 3D-cGA provides larger neighbourhood sizes than a 2D-cGA considering similar population sizes [28]; this is a consequence of the topology's dimension. Interconnection between the cells results in vertical expansion instead of horizontal expansion in a 2D grid. Although the interconnection leads the algorithm to be more exploitative, the balance between exploitation and exploration is kept by choosing an appropriate neighbourhood radius with respect to the grid topology [33]. If the selection pressure is dynamically controlled through these parameters, higher hit rates and better convergence times would be achieved.

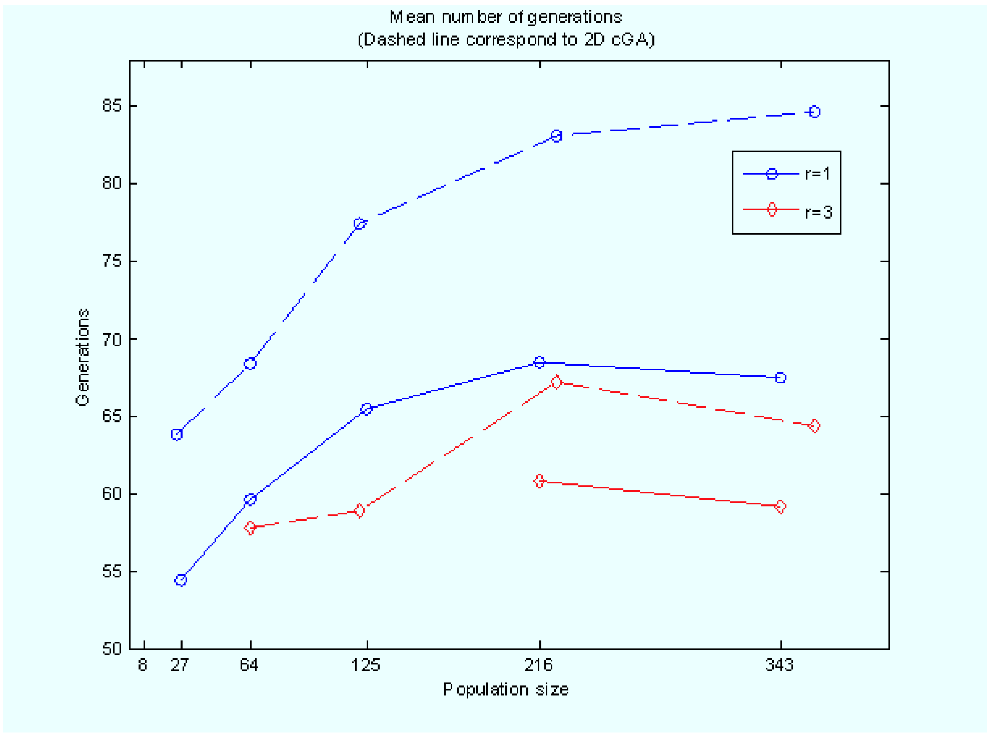


Figure 5.6: Ackley function average number of generations

In [94] further 3D-cGAs results were presented implementing a probabilistic local selection method as a dynamic control for selection pressure. In [95] an automatic 3D-cGA fault tolerant approach was proposed. The method considered the loss of genotypic diversity due to the faults, affecting the phenotypic space. Thus, an automatic isolation process was performed in order to allow non faulty individuals to migrate and to evolve within free-fault neighbourhoods. A variety of test problems were tackled and the proposed fault tolerant 3D-cGA was able to deal with up to 30% faulty individuals.

If the benefits of the performance results so far obtained are merged with the advantages that 3D technology has shown, the resulting architecture would offer significant advantages in terms of decreased routing length, reduced interconnections delay and increased logic and memory density. In the future it will be possible to improve today's optimization engines performance at algorithmic and implementation levels.

5.4 Chapter Summary

This chapter has focused on carrying out an empirical comparative analysis between 2D-cGAs and 3D-cGAs. The premise was that a 3D population topology would outperform

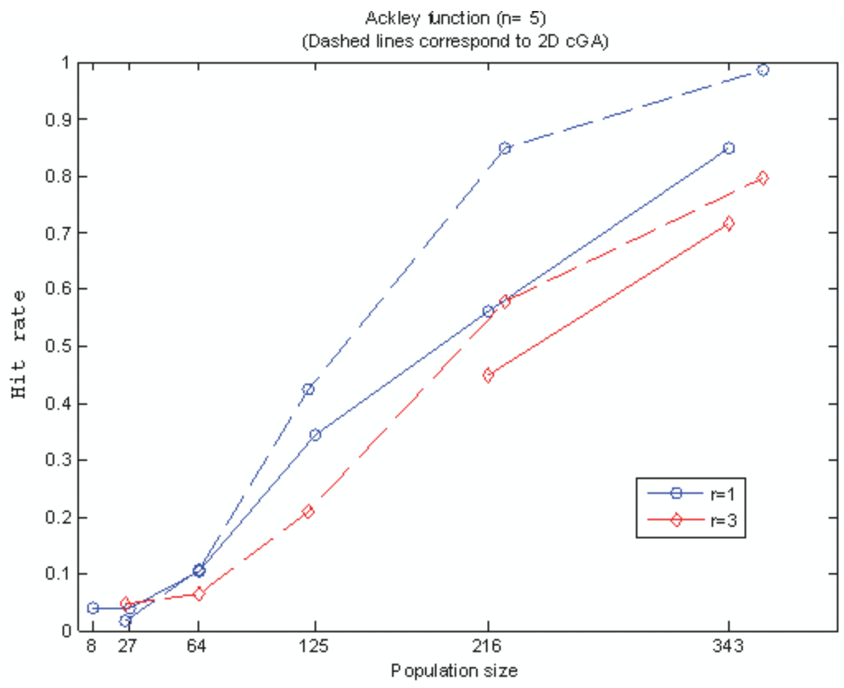


Figure 5.7: Ackley function hit rate

a 2D lattice configuration in algorithmic performance. A 3D structure presents shorter radii that allow faster solutions spreading and more dense local neighbourhoods that affects locally the induced selective pressure. The contribution to knowledge in this topic is:

- Having larger radius ($r = 3$) in a 2D topology shows the same growth rate for the best individual than shorter radius ($r = 1$) in a 3D topology. Thus, the same selective pressures are induced and similar NGRs are observed for different population sizes.

- Local neighbourhoods with the same radius present higher NGRs on 3D topologies while lower NGRs are shown in 2D topologies and therefore 3D-cGAs would perform a more exploitative search.

- Similar algorithmic performance in terms of convergence time (average number of generations to find the global optimum) and hit rates (number of successful experiments) are achieved by 2D and 3D topologies for different population sizes while tackling multi-modal and regular problems such as the Rastrigin and the Schwefel functions.

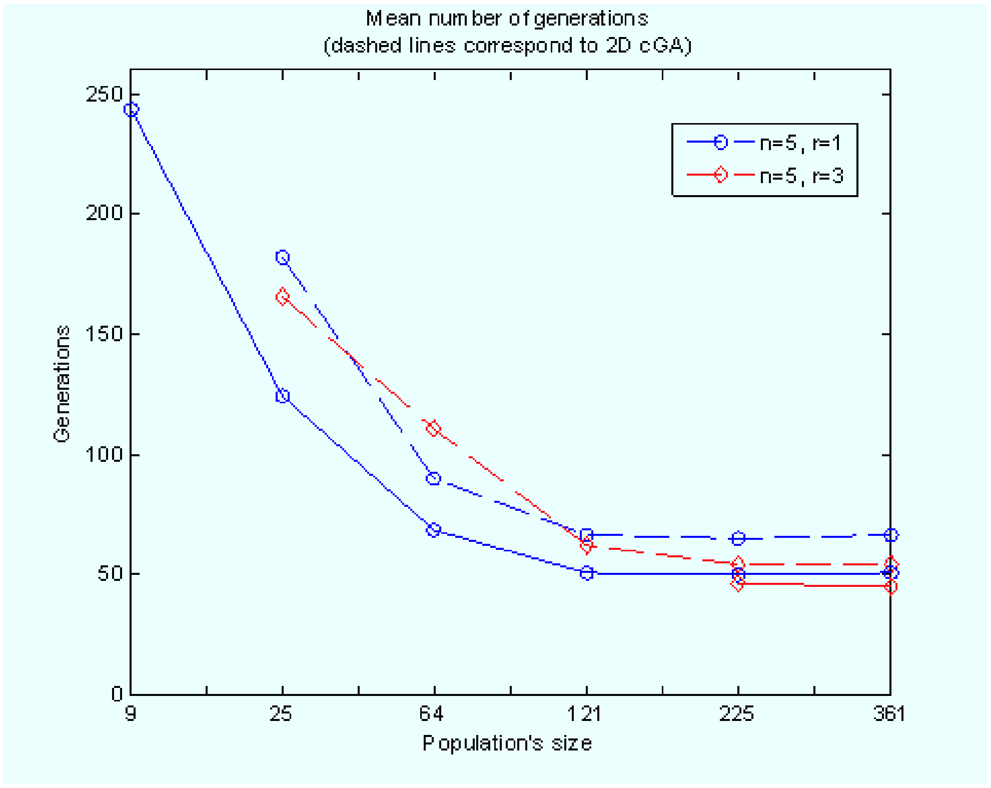


Figure 5.8: Griewank function average number of generations

- Better convergence times are achieved by 3D-cGAs when tackling multi-modal and epistatic problems such as the Griewank and Ackley functions. However, for the Ackley function the improvement is more noticeable than in the Griewank function. Although both functions present similar properties, the landscape of each individual problem present specific challenges for the search process.
- In cGAs exploration is carried out globally throughout the grid in concurrence with the exploitation of solutions that is promoted locally within neighbourhoods. 3D-cGAs provides denser neighbourhoods which allow more diversity for the local selection of solutions modifying implicitly the selection intensity. Moreover, the overall 3D structure present shorter radii for the fast spreading of solutions and these unique properties allow a better balance of the exploration-exploitation trade-off.

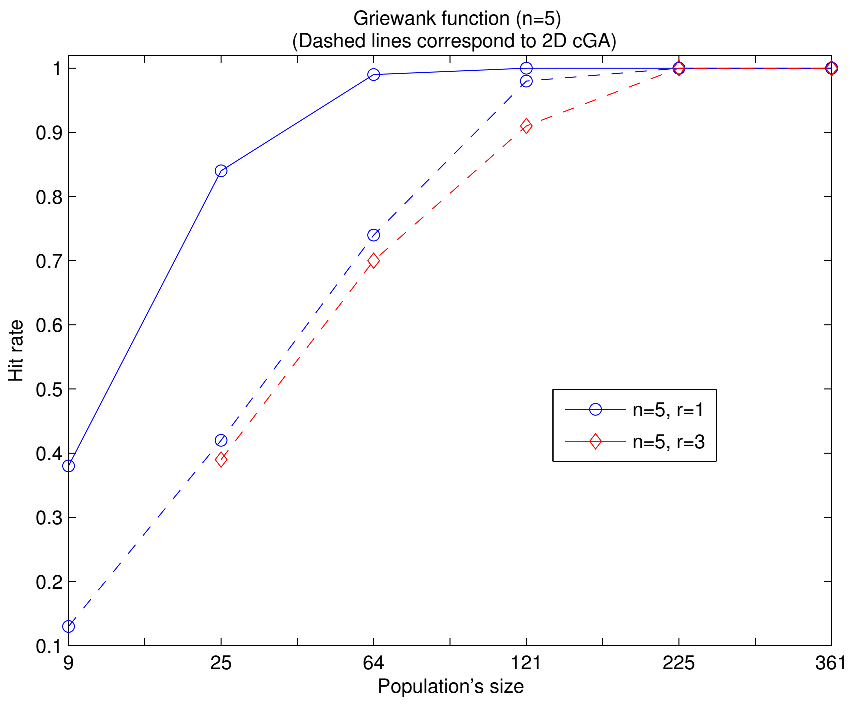


Figure 5.9: Griewank function hit rate

Chapter 6

Summary and Conclusions

The main aim of this thesis has been to investigate the implicit abilities of cellular Genetic Algorithms to deal with hard optimization problems. On the one hand, the fault tolerant arena is targeted through the usual genetic operations as well as the structural properties of this evolutionary technique functioning as mitigation techniques to overcome faulty scenarios, and thus successfully converge to the global optimum. On the other hand, cGAs structural properties are further investigated from a dynamic perspective in order to improve their performance. Measuring phenotypic and genotypic changes during evolution provides important information to guide the search while maintaining an adequate exploitation/exploration trade-off. For this purpose, an extended test bench was considered including continuous, real-world and combinatorial problems.

This chapter presents a summary of the work carried out in each chapter of this thesis, followed by the conclusions derived from the research presented. Finally, future research guidelines are provided.

6.1 Thesis Summary

Cellular Genetic Algorithms have successfully tackled difficult optimization problems that at different levels present characteristics such as multi-modality, epistasis and asymmetry, among others. In Subsection 2.4.4, an empirical study has been presented, comparing two widely used standard versions of GAs known as generational and steady state GAs and cellular GAs. Tight limits in the number of generations were defined, which might be a disadvantage for standard GAs; yet cGAs proved to outperform standard GAs under these restrictive experimental constraints. Limiting the maximum number of generations in the experimental set-ups evaluated in this thesis has been consistent with the hard real-time application arena considered during the development of this research.

Chapter 3 focuses on the fault tolerant arena. As it was more appropriate to follow a combined hardware and software approach to deal with SEEs, several mitigation techniques were proposed and were evaluated while targeting the hard real-time application known as the GPS attitude determination problem. This problem is highly multi-modal, epistatic and non regular, and therefore difficult to optimize. Moreover, if a GPS attitude system were to operate as part of the electronics for vehicle navigation, it would be subjected to harsh environmental conditions, such as in aircraft where electronics are prone to radiation; in such a case, if the essential data of the system were to be affected by SEEs, there would be a failure in an architecture implementing a cellular GA to calculate the attitude parameters. In this respect, two faulty scenarios were empirically assessed: SHEs or SEUs, both in their single and multiple bit versions, affecting fitness scores and chromosome registers.

SHEs and SEUs at the phenotypic space would radically affect the normal operation of the GPS attitude architecture. Indeed, stagnation or convergence to a local optima could be the search outcome. Applying normal genetic operations such as migration, or using cGAs inherent properties as mitigation techniques was demonstrated to be a viable tool to deal with faults. Explicit migration was implemented following several selection and replacement criteria and the cGAs ability to overcome faulty scenarios was demonstrated experimentally. On the other hand, the induced selection pressure was modified through changing the configuration of the local neighbourhood, using different sizes and shapes. In this way, faulty individuals have less opportunity for selection and therefore the search can still converge to the global optimum. In both cases, cGAs were able to deal with up to 40% stuck at zero faults, with up to 30% of stuck at one SHEs or SEUs and with similar percentages in the hybrid case, stuck at zero and one faults.

In the second part of Chapter 3, the fault tolerant approach consisted in injecting SHEs/SEUs into cells where chromosomes are allocated. The loss of diversity at the genotypic space is measured and the selection pressure is changed through the configuration of the population topology in order to induce a more explorative or more exploitative search and thus maintain an adequate diversity level in order to allow non-faulty individuals to continue evolving towards the global optimum. The same types of faults as in the phenotypic case were injected: stuck at zero, stuck at one and hybrid stuck at zero and one, considering in the worst case scenario up to 40% of chromosome bits to be faulty. Results reported an improvement in terms of results accuracy while the hit rate and convergence time were similar for a standard cGA and the proposed adaptive approach.

In the next section of the same chapter, a distributed parallel cellular GA was proposed to improve results accuracy compromised in previous reported results [5, 51]. Final observations pointed to the replacement policy as the main cause of losing results accuracy when the population size was increased. However, a parallel cGA with a non adequate replace-always policy still showed an improvement in the overall performance of the algorithm. A loose connection among cellular sub-structures was implemented where only individuals located at corners communicate with their adjacent neighbours, reducing the communication cost. Migration among sub-cellular structures based on phenotypic diversity changes showed the best convergence times and hit rates.

A fault tolerance perspective was then studied considering a distributed parallel cGA. SHEs and SEUs affecting fitness score registers were induced. A tuning parameter that triggers migration was evaluated considering flexible/middle/strict criteria for corner individuals to communicate with adjacent cellular sub-structures. Results showed a distributed parallel cGA outperformed a cGA, when stuck at zero faults occur in 25% of individuals. However, the proposed approach could only deal with up to 10% of stuck at one faults at fitness score registers. In general, cGAs demonstrated that they possess inherent abilities to deal with these kinds of faults affecting critical data in cGA based architectures.

Chapter 4 focuses on further investigating the dynamics of cellular GAs to deal with hard optimization problems. An extended test bench was therefore utilised including continuous, real-world and combinatorial problems. Two main experimental set-ups were proposed and assessed: 1) from a single-static to several-dynamic lattice topologies, 2) comparing *internal* versus *external* lattice reconfiguration.

Taking advantage of the inherent properties cellular GAs possess, either to deal with faulty scenarios or to improve their algorithmic performance, has been a consistent approach during the development of this research. The decentralized and structured population of cGAs is the main difference between them and standard or panmictic GAs. Dynamically modifying the lattice's configuration provides different levels of selection pressure that help in maintaining an adequate exploration/exploitation trade-off; which in many cases leads to a better overall performance. This was demonstrated in the first experimental set-up where different configuration topologies were gradually introduced. The lattice reconfiguration effect was assessed through two local selection methods presenting opposite selection pressures. Although statistically supported results for most of the problems were obtained when reconfiguration was carried out between two and three different grid configurations, the most significant improvements were achieved while having four lattice configuration options and a weak selection pressure induced locally through anisotropic selection.

The second experimental set-up aimed to directly compare the *external* reconfiguration approach proposed by Dorronsoro et al. with the *internal* lattice reconfiguration mechanism applied in the first experimental set-up. The *external* lattice reconfiguration approach necessarily implies the re-allocation of individuals and therefore an explicit form of migration is induced. Reconfiguring the grid *internally* helps in maintaining the original adjacency of individuals while increasing the inner selection pressure at cellular sub-structures. Thus, any improvement in algorithmic performance is only a consequence of the lattice reconfiguration mechanism. Statistically supported results on performance improvement were achieved in most of the problems in terms of convergence time and hit rate. The accuracy of the results was assured by using highly accurate thresholds.

In both experimental set-ups the dynamic reconfiguration mechanisms were performed constantly, every certain number of generations, or adaptively, based on phenotypic or genotypic diversity changes. One of the main conclusions of both set-ups is that no statistical difference was found in terms of convergence time between constant and adaptive approaches for the majority of the problems. It is however more convenient to use a constant reconfiguration scheme rather than adding computational cost by calculating the phenotypic or genotypic diversity.

In the final section of this chapter, another approach to dynamically controlling the selection pressure in cGAs is studied. A local selection method known as anisotropic selection introduced by Simoncini et al. was investigated as a control mechanism. Anisotropic selection also makes use of cGAs structural properties, particularly of the local neighbourhood configuration. It assigns selection probabilities to individuals according to their position in the neighbourhood; this induces different levels for selection pressure because not only the best (high selection pressure) or worst (low selection pressure) individuals are selected for reproduction. This approach was compared to the dynamic lattice reconfiguration mechanism presented before. In most problems, performance results were similar or in some cases better than dynamically reconfiguring the grid. Yet, for some problems (the Griewank function and the SLE) applying constant high selection pressure reported the best performances.

In Chapter 5, a comparative study of 2D and 3D cGAs is presented. Cellular GAs are normally implemented on 2D toroidal lattices; however, 3D toroidal structures present characteristics such as reduced radii for individuals' placement and thus shorter times for diffusion of solutions throughout the grid. A neighbourhood with more individuals located at the same distance implicitly induces the same selection pressure as in a 2D local neighbourhood with triple the radius. Several population sizes and neighbourhood configurations were empirically evaluated. 3D-cGAs achieved better convergence times when tackling multi-modal and epistatic problems. However, higher hit rates were only obtained for populations with less than a hundred individuals.

In future, optimization engines would combine the benefits at algorithmic level of 3D-cGAs with the latest improvements in 3D-IC technology, such as reduced routing length with consequently shorter interconnection delays and increasing memory and logic density, among others.

6.2 Conclusions

This thesis has further investigated the inherent abilities of cellular Genetic Algorithms as optimization engines while tackling difficult optimization problems. The fundamental idea has been to take advantage of specific cGAs' characteristics such as the structured and decentralized population, the implicit migration due to the neighbourhoods overlapping and thus the smooth diffusion of solutions throughout the grid, the explicit migration, the size and shape of the population topology and the local neighbourhood, the selection pressure induced locally by the selection method, etc.

At the initial stages of this research, a hard real-time application was tackled, the GPS attitude determination problem, a difficult optimization task presenting characteristics such as multi-modality, epistasis and non regularity. A fault tolerant approach was proposed to deal with SEE, in particular with SHEs and SEUs affecting data critical to the system such as fitness scores and chromosomes storage cells. The ability of cGAs to deal with faulty scenarios was empirically demonstrated, while applying explicit migration or modifying the selection pressure through the configuration of the topology or the local neighbourhood. Several migration policies as well as structural configuration criteria were tested. The proposed mitigation techniques are based on implicit characteristics of cGAs, therefore the added computational cost was minimal.

Following on from this, a dynamic approach based on cGAs structural properties was developed aiming to improve their overall performance. A wide variety of problems were tackled including continuous, real-world and combinatorial. Having a structured and decentralized population is a main characteristic of cGAs. It makes it possible to parallelize the search, while performing, at two levels, the exploitation and exploration of the solutions space. Exploitation is carried out locally through the neighbourhoods, and the exploration of the search space is performed globally throughout the grid. For most of the test problems, it was empirically demonstrated that through dynamically modifying the internal configuration of the population topology a significant improvement in performance could be achieved. These results were statistically supported.

A second approach was developed to dynamically modify the induced selection pressure through a local selection method known as anisotropic selection. This method is also based on the structural properties of cGAs, in particular the neighbourhood configuration. The extended test bench presented had been assessed through the proposed dynamic local selection approach, see Appendix A. A comparative analysis between both dynamic approaches: local selection versus lattice reconfiguration was carried out. Dynamic local selection achieved similar or in some cases better performances. Dynamics in both mechanisms were performed constantly or adaptively, based on phenotypic and genotypic diversity changes. Although detecting changes in both spaces provides a more accurate timing for dynamically changing the population topology, the effect in the overall performance of the adaptive criteria was not as significant as expected. Performing a constant reconfiguration of the grid every certain number of generations showed similar improvements with no added computational cost.

Finally, a comparative analysis of cGAs dimensionality was developed. 3D-cGAs achieved better performance in terms of convergence time than 2D-cGAs when tackling multi-modal and epistatic problems. However, only in small population sizes did 3D-cGAs outperform 2D-cGAs in terms of hit rates. This initial study was the starting point for ongoing research within the System Level Integration group. In future, it will be possible to combine the algorithmic advantages of 3D algorithmic structures and the latest improvements in 3D-IC technology.

6.3 Future Work

This thesis has thoroughly investigated cellular GAs, emphasizing the inherent abilities they possess to improve their performance. However, several interesting aspects remain open for further investigation which might contribute to current knowledge of cGAs as optimization engines.

In the fault tolerant arena, the proposed mitigation techniques can be extended and combined. For example, to consider SHEs and SEUs affecting both the phenotypic and genotypic spaces simultaneously. Analysing the consequence of such a scenario would be important. Genotypic changes are normally expressed at the phenotypic space. Yet, the opposite case is not always sustained. In the genotypic space, developing an isolation criteria for faulty individuals is an aspect open for further investigation. Before isolation, faulty individuals should be detected. A direct indication of faulty bits in chromosome registers is reflected by the loss of diversity in genotypes and consequently an incorrect mapping in the phenotypic space. However, the fact that phenotypes are faulty does not necessarily reflect on their chromosomes, and therefore faults in phenotypes should be detected in their own space. In this research, worst case scenarios were considered. However, more specific criteria for faults detection in the phenotypic space need to be proposed. Other internal structures of the cellular GA based architecture should also be explored from a fault tolerant perspective, for example the Finite-State Machine (FSM) that rules transitions between stages of evolution, among others.

It is demonstrated that a more adequate balance of the exploitation/exploration trade off is achieved through structurally modifying the population topology or the local selection; thus leading to an improvement in cGAs performance. A more refined approach is desirable for the formation of internal cellular substructures. In [96] a self-organizing topology EA is compared with a 1D-cGA implemented with certain automatic rules for dynamic topological changes. In this respect, on-the-fly information of genotypic and phenotypic diversity could provide some guidelines for limiting the borders of internal structures. Similarly, at local level the probability of neighbours for selection, assigned by the direction of individuals' position in anisotropic selection, can also be tuned automatically [30]. A combination of both is an immediate step; yet a careful analysis should be carried out to avoid an undesirable overbalance. The concurrent application of a global and a local dynamic control for selective pressure could deteriorate the natural course of individuals' evolution and could negatively affect cGAs performance.

An assessment from the fault tolerance point-of-view of the dynamic criteria for diversity tuning proposed in Chapter 4 is an immediate further extension of present work. The effect of SHEs or SEUs in chromosomes and fitness score registers or memory allocations directly influence diversity in phenotypic and genotypic spaces, therefore a structural based mechanism for an improved balance of the exploration - exploitation trade-off would benefit cGAs performance in such faulty scenarios. Dynamic anisotropic selection allocates probabilities for selection according to individuals position in local neighbourhoods, directly affecting the search's exploitation which is mainly promoted locally within neighbourhoods. This dynamic criterion can apply to the isolation process of possible faulty individuals from a local perspective.

Appendix A

Benchmark Problems

Dynamic cGAs proposed in this research are assessed through ten benchmark problems in the continuous, real-world and combinatorial domains. Five widely known functions presenting difficult landscapes with characteristics such as multi-modality, epistasis and non-symmetry are targeted in Chapters 4 and 5. Moreover, three real-world problems and three combinatorial problems are also evaluated. In the following section, the characteristics of those problems are detailed.

A.1 Continuous Problems

Continuous problems are most commonly evaluated in the real domain [97]. However, due to binary chromosomes encoding, continuous functions have been targeted here in the discrete domain with a specific minimum step per variable. An accurate average fitness score threshold has been assessed in all experimental cases. The aim is to evaluate difficult characteristics of theoretical continuous problems that are commonly found in real-world problems.

- Rastrigin function

$$f\left(\vec{x}\right) = 10n + \sum_{i=1}^{n} \left(x^2 - cos\left(2\pi x_i\right)\right) \tag{A.1}$$

  n is the dimension of the function and x is each encoded variable where $x_i \in (-5.12, 5.12)$. The global minima is located at $x_{min}\left(0, 0, ..., 0\right)$. The minimum step, per variable, in the encoded solution is $x_{min} = \frac{10}{2^{10}-1}$. An average fitness score threshold $\bar{f} \leq 0.0005$ is evaluated.

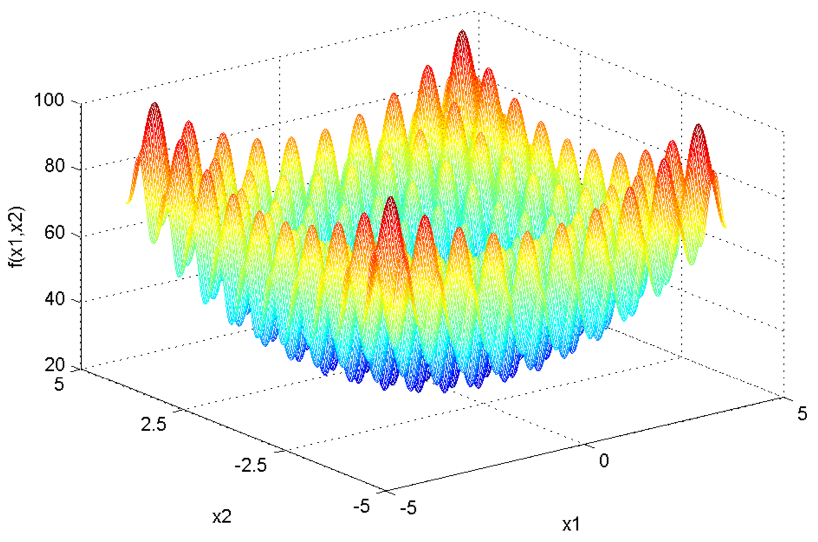


Figure A.1: Rastrigin function search space

Rastrigin function is highly multi-modal, regular, and separable and it is catalogued as difficult for most optimization techniques. Multi-modality means having several local optima in the search space. Multi-modality increases the difficulty of the problem, because the searching process needs to escape multiple local optima to avoid stagnation. In this function, local optima are distributed symmetrically which makes it regular. Function separability refers to genes inter-dependency; thus in a separable function, there is no epistatic effect among variables. Therefore, optimization tackles each variable independently.

- Schwefel function

$$f\left(\vec{x}\right) = 418.9829n + \sum_{i=1}^{q} x_i sin\left(\sqrt{|x_i|}\right) \tag{A.2}$$

q is the dimension of the function and x is each encoded variable where $x_i \in (-500, 500)$. The global minima is located at $x_{min}\,(420.9687, 420.9687, ..., 420.9687)$. The minimum step, per variable, in the encoded solution is $x_{min} = \frac{10^3}{2^{10}-1}$.

The Schwefel function is highly multi-modal, regular, and separable and it is catalogued as difficult for most optimization techniques [98], see Figure A.2.

- Griewank function

$$f\left(\vec{x}\right) = 1 + \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod cos\left(\frac{x_i}{\sqrt{i}}\right) \tag{A.3}$$

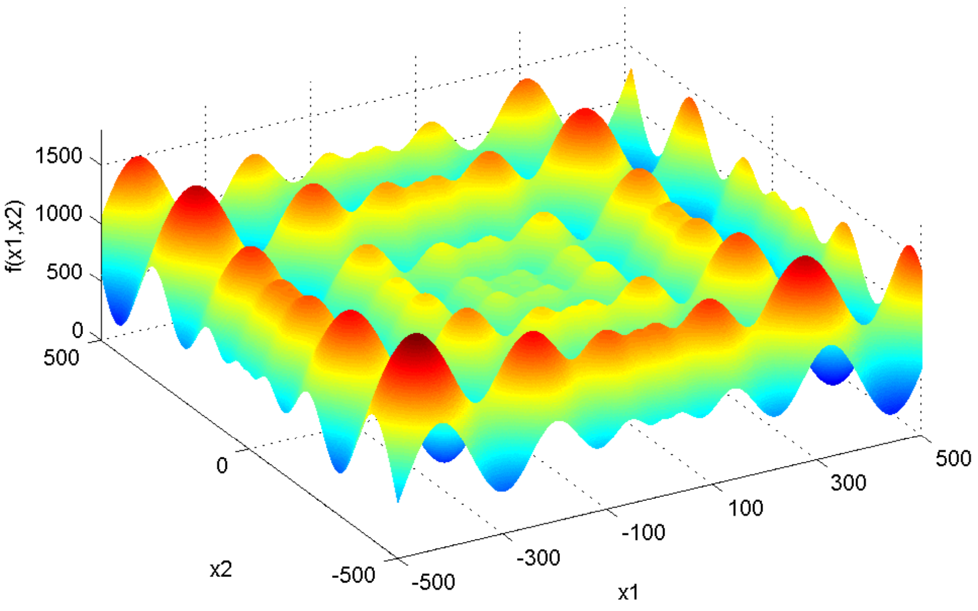


Figure A.2: Schwefel function search space

n is the dimension of the function and x is each encoded variable, with $x_i \in (-600, 600)$. The global minimum is $x_{min}(0, ...0)$. The minimum step, per variable, in the encoded solution is $x_{min} = \frac{1\times10^3}{2^{10}-1}$. An average fitness score threshold $\bar{f} = 0.0001$ has been assessed in all experimental cases.

As the number of dimensions increases the number of minima grows exponentially. Griewank function is multi-modal, regular, and non-separable. Non-separable functions are epistatic. Epistasis defines genes interdependency through the modification of one gene by one or more genes. Therefore, this kind of function is more difficult to optimize since moving from one point to another in the search space highly depends on the joint action of two or more genes.

- Ackley function

$$f(\vec{x}) = 20 + e^{-20\left(-0.2\sqrt{\frac{1}{q}\sum_{i=1}^{q} x_i^2}\; e^{\left(\frac{1}{q}\sum_{i=1}^{q} cos(2\pi x_i)\right)}\right)} \quad (A.4)$$

q is the dimension of the function and x is each encoded variable where $x_i \in (-30, 30)$. The global minima is located at $x_{min}(0, 0, ..., 0)$. The minimum step, per variable, in the encoded solution is $x_{min} = \frac{60}{2^{10}-1}$. The Ackley function is highly multi-modal, regular, and non-separable, see Figure A.4.

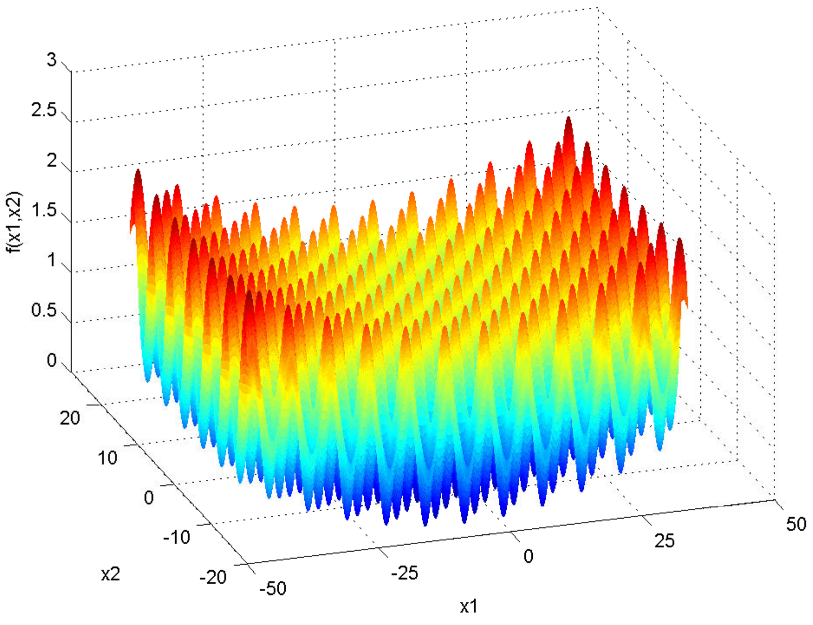


Figure A.3: Griewank function search space

- Langerman function

$$f(\vec{x}) = -\sum_{i=1}^{n} c_i e^{-\frac{1}{\pi}\sum_{j=1}^{D}\left(x_j - a_{ij}^2\right)} cos\left(\pi\sum_{j=1}^{D}\left(x_j - a_{ij}\right)^2\right) \tag{A.5}$$

n is the dimension of the function and x_i is each encoded variable, with $x_i \in (0, 10)$. The minimum step, per variable, in the encoded solution is $x_{min} = \frac{10}{2^{10}-1}$. Similar to previous test functions, Langerman function is highly multi-modal and epistatic but also presents a random distribution of its local minima. Thus, this function presents no advantages in symmetry due to local optima distribution and can be considered the hardest one of these three problems. For testing purposes, matrix a_{ij} and vector c_i were taken from [99].

A.2 Real-world Problems

Three real problems have been evaluated. The Frequency Modulation Sound (FMS) and the System of Linear Equations (SLE) problems are selected from [82]; the third is the GPS attitude determination problem tackled in the previous chapter from a fault tolerant perspective, details for this problem are included in Chapter 3, Section 3.1. These problems present highly multi-modal and highly epistatic landscapes; while the

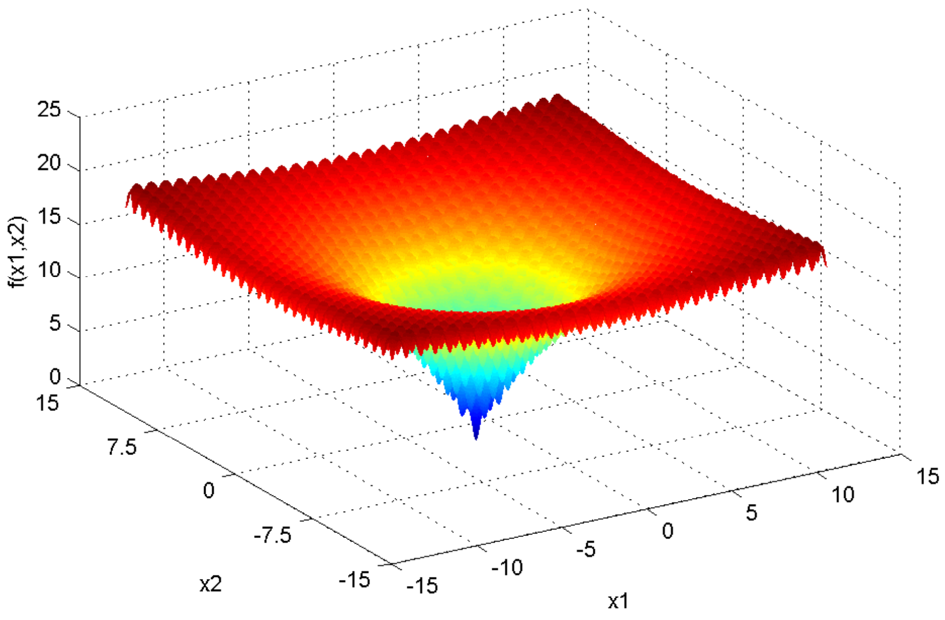


Figure A.4: Ackley function search space

FMS and the GPS problems have also a non-symmetrical distribution of their local and global optima.

- Frequency Modulation Sound Problem (FMS)

  The FMS parameter identification problem consists in finding the corresponding real parameters $(a_1, w_1, a_2, w_2, a_3, w_3)$ of:

$$\begin{aligned} f(t) = {} & a_1 \cdot sin(w_1 \cdot t \cdot \theta \\ & + a_2 \cdot sin\left(w_2 \cdot t \cdot \theta + a_3 \cdot sin\left(w_3 \cdot t \cdot \theta\right)\right)) \end{aligned} \tag{A.6}$$

  which fit the sound wave given by:

$$\begin{aligned} f(t) = {} & 1.0 \cdot sin(5.0 \cdot t \cdot \theta \\ & - 1.5 \cdot sin\left(4.8 \cdot t \cdot \theta + 2.0 \cdot sin\left(4.9 \cdot t \cdot \theta\right)\right)) \end{aligned} \tag{A.7}$$

  where $\theta = 2\pi/100$ and $w_{1,2,3}$ and $a_{1,2,3} \in [-6.4, 6.35]$. To minimize the sum of square errors is the aim of this problem which is highly multi-modal and epistatic, as observed in the equations above.

- System of Linear Equations (SLE)

  Solving a matrix equation of the form $Ax = b$ is the aim of this problem. A and

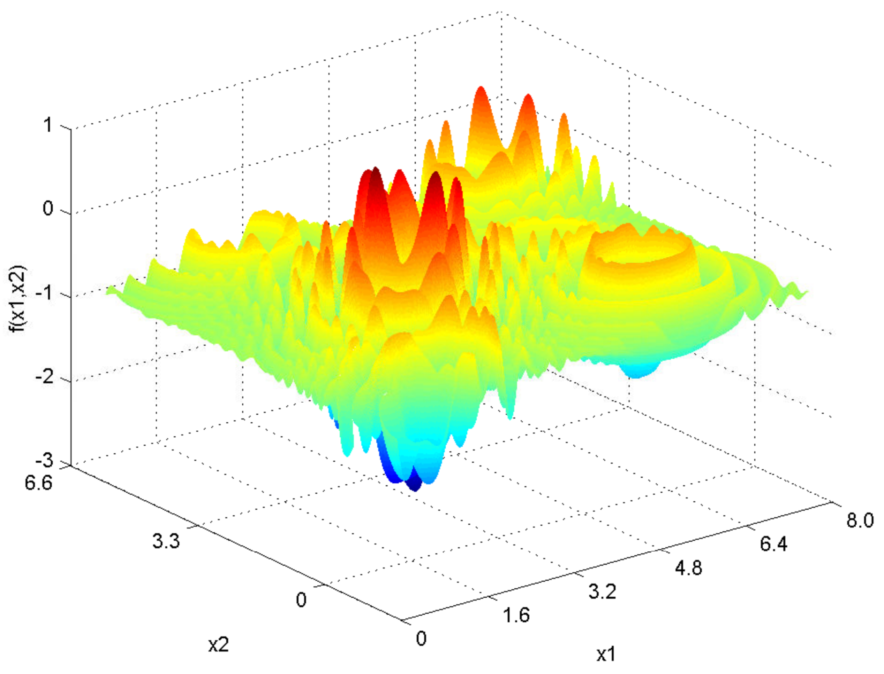


Figure A.5: Langerman function search space

b matrices are the same to those used in [82]. These are given by:

$$A = \begin{pmatrix} 5 & 4 & 5 & 2 & 9 & 5 & 4 & 2 & 3 & 1 \\ 9 & 7 & 1 & 1 & 7 & 2 & 2 & 6 & 6 & 9 \\ 3 & 1 & 8 & 6 & 9 & 7 & 4 & 2 & 1 & 6 \\ 8 & 3 & 7 & 3 & 7 & 5 & 3 & 9 & 9 & 5 \\ 9 & 5 & 1 & 6 & 3 & 4 & 2 & 3 & 3 & 9 \\ 1 & 2 & 3 & 1 & 7 & 6 & 6 & 3 & 3 & 3 \\ 1 & 5 & 7 & 8 & 1 & 4 & 7 & 8 & 4 & 8 \\ 9 & 3 & 8 & 6 & 3 & 4 & 7 & 1 & 8 & 1 \\ 8 & 2 & 8 & 5 & 3 & 8 & 7 & 2 & 7 & 5 \\ 2 & 1 & 2 & 2 & 9 & 8 & 7 & 4 & 4 & 1 \end{pmatrix} ; \; b = \begin{pmatrix} 40 \\ 50 \\ 47 \\ 59 \\ 45 \\ 35 \\ 53 \\ 50 \\ 55 \\ 40 \end{pmatrix}$$

Evolutionary techniques and particularly GAs are not the most appropriate optimization tool to solve a system of linear equations, linear programming is more suitable for this aim [1]. However, an analysis of the performance the proposed dynamic criteria achieve is still feasible, other authors have also tackled a SLE through GA such as in [12, 82].

A.3 Combinatorial Problems

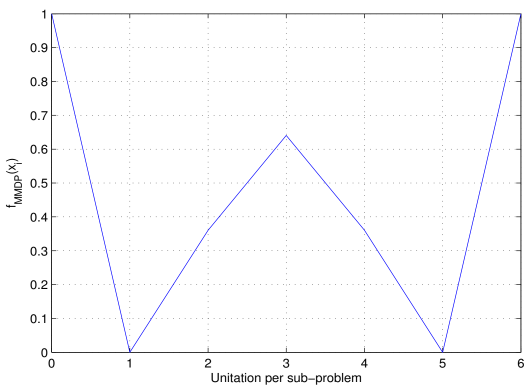
Combinatorial problems are discrete domain problems. The solution to the problem is directly represented in the encoded individual. Three difficult combinatorial problems are studied and evaluated in this thesis.

- Massively Multimodal Deceptive Problem (MMDP)

  MMDP is a problem composed by q sub-problems. The fitness value of each sub-problem reflects the number of ones (*unitation*) each sub-problem has. A very simple lookup table with assigned values is used, see Table A.1. The number of local and global optima would depend on the size of the problem. In this research, a size of $q = 25$ sub-problems has been used. Therefore the fitness function will sum up individual fitness per sub-problem (x) and a value of 25 will be obtained when the global optimum is reached. The fitness function is given by

$$f_{MDDP}(\vec{x}) = \sum_{i=1}^{q} fitness_{x_i} \tag{A.8}$$

Table A.1: MMDP lookup table



Ones	$f_{MMDP}\,(x_i)$
0	1.0000
1	0.0000
2	0.3603
3	0.6405
4	0.3603
5	0.0000
6	1.0000

  In Table A.1, the graph indicates that each sub-problem has a deceptive point in the middle and two global maxima at the extremes. This problem presents a large number of local optima in comparison to the number of global ones which is 2^q, where q is the number of sub-problems [82].

- Minimum Tardy Task Problem (MTTP)

  MTTP is a NP-complete combinatorial optimization problem. An optimum solution should fulfil corresponding schedule constraints. In the objective function, an encoded solution that does not fulfil scheduling constraints is penalized. This problem is highly multi-modal and epistatic. An instance of this problem is generated considering a MTTP scalable test problem introduced in [100]. A problem size of $p = 100$ tasks has been targeted with a population size of 100 individuals, with which an adequate performance to evaluate the proposed dynamic lattice reconfiguration mechanisms is achieved.

To generate large problem sizes, a minimum MTTP instance with $p = 5$ tasks, shown in Table A.2, has been scaled to generate a problem size of $p = 100$ tasks.

Table A.2: MTTP problem instance

task	1	2	3	4	5
length	3	6	9	12	15
deadline	5	10	15	20	25
weight	60	40	7	3	50

Scheduling constraints are:

– A task (x) cannot be scheduled before the last scheduled task had finished.

– Tasks must finish according to deadlines.

A scheduling function $g(\vec{x})$ for $i = 1, ..., p$ tasks is defined to fulfil those constraints. The fitness function includes the weight of each task which cannot be scheduled as a measure of feasibility. Thus, better solutions are less penalized. The fitness function is expressed in terms of the corresponding tasks' weights as

$$f_{MTTP}(\vec{x}) = \sum_{i=1}^{p} weight_{x_i} \tag{A.9}$$

- P-Peaks Problem

The P-Peaks problem is a tunable test problem that can vary its epistasis level if the number of generated peaks is increased or decreased. Every instance of the problem is different, thus it is also a non symmetric problem due to the random location of its solution [101]. P binary strings are randomly generated with Q bits of length representing the location of the peaks. Increasing or decreasing the number of peaks defines the difficulty of the problem and therefore the level of epistasis. In this thesis, for testing purposes the number of peaks is set to

$P = 100$ with binary strings of $Q = 100$ bits of length. Thus, the fitness score of an individual is calculated as the Hamming distance between the individual and the closest peak. Normalizing the calculated distance provides the fitness score. Hence, the maximum fitness score in the following fitness function is equal to 1.0.

$$f_{P-Peaks}(\vec{x}) = \frac{1}{Q} max_{i=1}^{P} \left(Q - Hamming\left(\vec{x}, P_i\right)\right) \tag{A.10}$$

Appendix B

List of Publications

1. **A. Morales-Reyes**, E. F. Stefatos, A. T. Erdogan, T. Arslan, ”Fault tolerant cellular genetic algorithm”, In Proceedings of the *IEEE Congress on Evolutionary Computation*, p.2676-2682, June 2008, Hong Kong.

2. **A. Morales-Reyes**, E. F. Stefatos, A. T. Erdogan, T. Arslan, ”Towards fault tolerant systems based on cellular genetic algorithms”, In Proceedings of the *NASA/ESA Conference on Adaptive Hardware and Systems*, pp. 398-405, June 2008, Noordwijk, The Netherlands.

3. **A. Morales-Reyes**, N. Haridas, A. T. Erdogan, T. Arslan, ”Adaptive Fault tolerant GPS Attitude Determination System”, In Proceedings of the 2009 *IEEE Aerospace Conference*, pp., March 2009, Big Sky, Montana, US.

4. **A. Morales-Reyes**, A. T. Erdogan, T. Arslan, ”A Distributed Cellular GA Based Architecture for Real Time GPS Attitude Determination”, In Proceedings of the 2009 *IEEE Congress on Evolutionary Computation*, pp., 18th-21st May 2009, Trondheim, Norway.

5. **A. Morales-Reyes**, A. Al-Naqi, A.T. Erdogan, T. Arslan, Towards 3D Architectures: A Comparative Study on Cellular GAs Dimensionality, *NASA/ESA Conference on Adaptive Hardware and Systems*, pp.223-229, San Francisco, California, 29th July, 2009.

6. **A. Morales-Reyes**, A.T. Erdogan, T. Arslan, Lattice Reconfiguration vs. Local Selection Criteria for Diversity Tuning in Cellular GAs, the 2010 *IEEE Congress on Evolutionary Computation*, Barcelona, Spain, July 1823, 2010.

7. **A. Morales-Reyes**, A.T. Erdogan, T. Arslan, Internal Lattice Reconfiguration for Diversity Tuning in Cellular Genetic Algorithms, submitted *IEEE Transactions on Evolutionary Computation.*

Bibliography

[1] D. A. Pierre, *Optimization Theory with Applications.* DOVER. 2nd. Edition, 1986.

[2] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley Publishers, 1989.

[3] E. Cantu-Paz, *Efficient and Accurate Parallel Genetic Algorithms.* Genetic Algorithms and Evolutionary Computation. Kluwer Academic Publishers, 2000.

[4] F. Lobo, C. Lima, and Z. Michalewicz, *Parameter Setting in Evolutionary Algorithms.* Springer. 1st. Edition, 2007.

[5] J. Xu, T. Arslan, D. Wan, and Q. Wang, "Gps attitude determination using a genetic algorithm," in *Proceedings of IEEE Congress on Evolutionary Computation, CEC'02.* IEEE, 2009, pp. 998–1002. [Online]. Available: ieeexplore.ieee.org/

[6] E. Alba and B. Dorronsoro, "The exploration/exploitation tradeoff in dynamic cellular genetic algorithms," in *IEEE Transactions on Evolutionary Computation.* IEEE, 2005.

[7] F. Rothlauf, *Representation for Genetic and Evolutionary Algorithms.* Springer. 2nd. Edition, 2006.

[8] R. L. Haupt and S. E. Haupt, *Practical Genetic Algorithms.* Willey-Interscience. 2nd. Edition, 2004.

[9] L. Sekanina, *Evolvable Components: From Theory to Hardware Implementations*, 1st ed. Springer-Verlag Berlin and Heidelberg GmhB & Co. K, October 2003.

[10] T. Bäck, *Evolutionary Algorithms in Theory and Practice.* Oxford University Press, 1996.

[11] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs.* Springer. Series: Artificial Intelligence, 1996.

[12] E. Alba, F. Luna, and A. Nebro, "Advances in parallel heterogeneous genetic algorithms for continuous optimization," in *International Journal of Applied Mathematics and Computer Science.* AMCS, 2004, pp. 14(3):101–117.

[13] E. Alba and M. Tomassini, "Parallelism and evolutionary algorithms," in *IEEE Transactions on Evolutionary Computation.* IEEE Press, 2002, pp. 6(5):443–462.

[14] E. Alba and J. M. Troya, "A survey of parallel distributed genetic algorithms," *Complex.*, vol. 4, no. 4, pp. 31–52, 1999.

[15] E. Cantu-Paz, "A summary of research on parallel genetic algorithms," in *IlliGAL report 95007, University of Illinois at Urbana-Champaign*, 1995.

[16] M. Tomassini, "Parallel and distributed evolutionary algorithms: A review," 1999.

[17] F. Herrera and M. Lozano, "Gradual distributed real-coded genetic algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 1, pp. 43–63, 2000.

[18] S. Baluja, "Structure and performance of fine-grain parallelism in genetic search," in *Technical Report, Carnegie Mellon University*, 1993.

[19] E. Alba, "Analysis and design of parallel distributed genetic algorithms," in *PhD Dissertation (in Spanish).* Universidad de Malaga, 1999.

[20] E. Cantu-Paz, "Migration policies and takeover times in parallel genetic algorithms," in *Proceedings of GECCO-99: Genetic and Evolutionary Computation Conference*, 1999, pp. 775–.

[21] E. Alba, A. Nebro, and J. Troya, "Heterogeneous computing and parallel genetic algorithms," in *Journal of Parallel and Distributed Computing.* Elsevier Science, 2002, pp. 62(9):1362–1385.

[22] M. Rebaudengo and M. S. Reorda, “An experimental analysis of effects of migration in paralell genetic algorithms,” in *IEEE/Euromicro Workshop on Parallel and Distributed Processing*, 1993, pp. 232–238.

[23] E. Cantu-Paz and D. E. Goldberg, “On the scalability of parallel genetic algorithms,” in *Evolutionary Computation by the Massachusetts Institute of Technology*, vol. 7, no. 4, 1999, pp. 429– 449.

[24] ——, “Topologies, migration rates, and multi-population parallel genetic algorithms,” in *GECCO-99: Genetic and Evolutionary Computation Conference*, vol. 7, no. 4, 1999, pp. 91–98.

[25] C. L. S. Park and J. Kim, “Topology and migration policy of fine-grained parallel evolutionary algorithms for numerical optimization,” in *IEEE Congress on Evolutionary Computation*, 2000, pp. 70–76.

[26] T. Bäck and R. Breukelaar, “Using genetic algorithms to evolve behavior in cellular automata,” in *Lecture Notes in Computer Sciences 3699*. Springer-Verlag, 2005, pp. 1–10.

[27] S. Olariu and A. Y. Zomaya, *Handbook of Bioinspired Algorithms and Applications*. Chapman & Hall/CRC. 1st. Edition, 2006.

[28] R. Breukelaar and T. Bäck, “Using a genetic algorithm to evolve behaviour in multi dimensional cellular automata,” in *Proceeding of GECCO 2005: Genetic and Evolutionary Computation Conference*. ACM, 2005, pp. 107–113.

[29] D. Simoncini, S. Verel, P. Collard, and M. Clergue, “Anisotropic selection in cellular genetic algorithms,” in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO’06)*, 2006, pp. 559 –566.

[30] ——, “Centric selection: a way to tune the exploration/exploitation tradeoff,” in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO’09)*, 2009, pp. 891 –898.

[31] J. Sarma and K. D. Jong, “An analysis of the effects of neighbourhood size and shape on local selection algorithms,” in *Parallel Problem Solving from Nature*. Springer, 1996, pp. 236–244.

[32] M. Tomassini, *Spatially Structured Evolutionary Algorithms, Artificial Evolution in Space and Time.* Springer. Series: Natural Computing Series, 2005.

[33] E. Alba and J. Troya, "Cellular evolutionary algorithms: Evaluating the influence of ratio," in *Proceeding of the 6th International Conference on Parallel Problem Solving from Nature.* LNCS Springer, 2000, pp. 29–38.

[34] P. Hansen and N. Mladenović, "Variable neighbourhood search: Principles and applications," *European Journal of Operational Research*, vol. 130, pp. 449–467, 2001.

[35] S. Consoli, K. Darby-Dowman, N. Mladenović, and J. A. M. Pérez, "Greedy randomized adaptive search and variable neighbourhood search for the minimum labelling spanning tree problem," *European Journal of Operational Research*, p. Submitted draft version, 2008.

[36] L. Liberti and M. Drazić, "Variable neighbourhood search for the global optimization of constrained nlps," L. N. in Operations Research by World Publishing Corporation (ISTP indexed), Ed. Proceeding of the Global Optimization Conference, 2005, pp. 1–5.

[37] A. Lusa and C. N. Potts, "A variable neighbourhood search algorithm for the constrained task allocation problem," *Journal of the Operational Research Society*, vol. 59, pp. 812–822, 2008.

[38] A. A. Sharov. (2010) Logistic model @ONLINE. [Online]. Available: http://home.comcast.net/ sharov/PopEcol/lec5/logist.html

[39] P. Spiessens and B. Manderick, "A massively parallel genetic algorithm," in *Proceedings of the 4th International Conference on Genetic Algorithms.* Morgan Kaufmann, 1991, p. 279286.

[40] E. Alba, "Parallel evolutionary algorithms can achieve super-linear performance," in *Information Processing Letters.* Elsevier, 2002, pp. 82(1):7–13.

[41] E. Alba and J. Troya, "Improving flexibility and efficiency by adding parallelism to genetic algorithms," in *Statistics and Computing.* Kluwer Academic Publishers, 2002, pp. 12(2):91–114.

[42] E. Alba, J. M. Troya, and J. M. Troya, "An analysis of synchronous and asynchronous parallel distributed genetic algorithms with structured and panmictic islands," in *In IPPS/SPDP Workshops*, 1999, pp. 248–256.

[43] E. Alba, C. Cotte, and J. M. Troya, "Numerical and real time analysis of parallel disributed gas with structured and panmictic populations," in *IEEE Transactions on Evolutionary Computation*. IEEE, 1999, p. 1026.

[44] ——, "Entropic and real-time analysis of the search with pancmictic, structured, and parallel distributed genetic algorithms," in *LCC Technical Report ITI 99-7*. Universidad de Malaga.

[45] B. Schöfisch and A. de Roos, "Synchronous and asynchronous updating in cellular automata," in *BioSystems*, vol. 51. Elsvier, 1999, pp. 123–143.

[46] M. Giacobini, M. Tomassini, A. Tettamanzi, and E. Alba, "Selection intensity in cellular evolutionary algorithms for regular lattices," in *IEEE Transactions on Evolutionary Computation*. IEEE, 2005, pp. 489–505.

[47] B. Turton, T. Arslan, and D. Horrocks, "A hardware architecture for a parallel genetic algorithm for image registration," in *IEE Colloquium on Genetic Algorithms in Image Processing and Vision*, October 1994, pp. 11/1–11/6.

[48] B. Turton and T. Arslan, "An architecture for enhancing image processing via parallel genetic algorithms and data compression," in *IEE/IEEE International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications*, September 1995, pp. 493–498.

[49] J. Evans and T. Arslan, "The implementation of an evolvable hardware system for real time image registration on a system-on-chip platform," in *NASA/DoD Conference on Evolvable Hardware*, July 2002, pp. 142–146.

[50] B. Turton and T. Arslan, "A parallel genetic vlsi architecture for combinatorial real-time applications - disc scheduling," in *IEE/IEEE International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications*, September 1995, pp. 493–498.

[51] J. Xu, T. Arslan, Q. Wang, and D. Wan, "An ehw architecture for real-time gps attitude determination based on parallel genetic algorithm," in *Proceedings of Conference on Evolvable Hardware NASA/DoD*, 2002, pp. 133–141.

[52] E. Stefatos and T. Arslan, "An efficient fault-tolerant vlsi architecture using parallel evolvable hardware technology," in *Proceedings of NASA/DoD Conference on Evolvable Hardware.* IEEE, 2004, pp. 97–103.

[53] ——, "High-performance adaptive gps attitude determination vlsi architecture," in *IEEE Workshop on Signal Processing Systems, SIPS.* IEEE, 2004, pp. 233–238.

[54] R. Thomson and T. Arslan, "An evolutionary algorithm for the multi-objective optimization of vlsi primitive operator filters," in *Congress on Evolutionary Computation*, May 2002, pp. 37–42.

[55] ——, "On the impact of modelling, robustness, and diversity to the performance of a multi-objective evolutionary algorithm for digital vlsi system design," in *Congress on Evolutionary Computation*, December 2003, pp. 382–389.

[56] ——, "Evolvable hardware for the generation of sequential filter circuits," in *NASA/DoD Conference on Evolvable Hardware*, July 2002, pp. 17–25.

[57] ——, "Techniques for the evolution of pipelined linear transforms," in *IEEE Congress on Evolutionary Computation*, vol. 3, September 2005, pp. 2476–2482.

[58] E. Stefatos, T. Arslan, and A. Hamilton, "Enhanced evolutionary techniques for precise and real-time implementation of low-power fir filters," in *IEEE Congress On Evolutionary Computation*, 2008.

[59] R. Andraka, "A survey of cordic algorithsm for fpga based computers." ACM, 1998, pp. 191–200.

[60] K. A. Label, "Single event effect criticality analysis," in *NASA Headquarters/ Code QW*, 1996.

[61] P. K. Lala, *Fault Tolerant & Fault Testable Hardware Design*, 1st ed. Prentice Hall International, 1985.

[62] R. Lyons and W. Vanderkulk, “The use of triple-modular redundancy to improve computer reliability,” in *IBM Journal*, 1962.

[63] T. Calin, M. Nicolaidis, and R. Velazco, “Upset hardening memory design for submicron cmos technology,” in *IEEE Transactions on Nuclear Science*, 1996, pp. 2874–2878.

[64] X. Iturbe, M. Azkarate, I. Martínez, J. Perez, and A. Astarloa, “A novel seu, mbu and she handling strategy for xilinx virtex-4 fpgas,” in *Proceedings of The International Conference on Field Programmable Logic and Applications.* IEEE, 2009, pp. 569–573.

[65] C. Constantinescu, “Trends and challenges in vlsi circuit reliability,” in *IEEE MICRO.* IEEE Computer Society, 2003, pp. 14–19.

[66] S. Hareland, J. Maiz, M. Alavi, K. Mistry, S. Walsta, and C. Dai, “Impact of cmos process scaling and soi on the soft error rates of logic processes,” in *Symposium on VLSl Technology Digest of Technical Papers.* IEEE, 2001, pp. 73–74.

[67] Xilinx, “Device reliability report,” in *UG116 (v5.8).* Xilinx, 2010.

[68] P. Adell and G. Allen, “Assessing and mitigating radiation effects in xilinx fpgas.” Jet Propulsion Laboratory, 2008.

[69] G. Greenwood, “On the practicality of using intrinsic reconfiguration for fault recovery,” in *IEEE Transactions on Evolutionary Computation*, vol. 9(4). IEEE, 2005, pp. 398–405.

[70] ——, “Attaining fault tolerance through self-adaptation: The strengths and weaknesses of evolvable hardware approaches,” in *2008 World Congress on Computational Intelligence, Plenary Invited Lecture*, J. Z. et al., Ed. LNCS, Springer-Verlag, 2008, pp. 368–387.

[71] J. Juang and G. Huang, “Development of gps-based attitude determination algorithms,” in *IEEE Transactions on aerospace and electronics systems*, 1997.

[72] M. Hodgart and S. Purivigraipong, “New approach to resolving instantaneous integer ambiguity resolution for spacecraft attitude determination using gps signals,” in *IEEE Position, Location and Navigation Symposium*, 2000, pp. 132–139.

[73] T. N. Limited. (2010) Gps tutorial @ONLINE. [Online]. Available: http://www.trimble.com

[74] G. Xu, *GPS Theory, Algorithms and Applications.* Springer. 2nd. Edition, 2007.

[75] Kowoma. (2009) The gps system @ONLINE. [Online]. Available: http://www.kowoma.de/en/gps/signals.htm

[76] A. Muhammad, A. Bargiela, and G. King, "Fine-grained parallel genetic algorithm: A stochastic optimisation method," 1990.

[77] ——, "Fine-grained parallel genetic algorithm: A global convergence criterion," in *International Journal in Computer Mathematics.* Gordon and Breach Science, 1999, pp. 139–155.

[78] S. Gordon, K. Mathias, and D. Whitley, "Cellular genetic algorithms as function optimizers: Locality effects," in *proceedings of the 1994 ACM Symposium on Applied Computing.* ACM Press, 1994, pp. 237 – 241.

[79] A. Morales-Reyes, E. Stefatos, A. Erdogan, and T. Arslan, "Fault tolerant cellular genetic algorithm," in *Proceedings of IEEE Congress on Evolutionary Computation, 2008. (CEC 2008).* IEEE, 2008, pp. 2671–2677. [Online]. Available: ieeexplore.ieee.org/

[80] ——, "Towards fault-tolerant systems based on adaptive cellular genetic algorithms," in *Proceedings of IEEE NASA/ESA Conference on Adaptive Hardware and Systems (AHS'08).* IEEE, 2008, pp. 398–405. [Online]. Available: ieeexplore.ieee.org/

[81] C. Oei, D. Goldberg, and S.-J. Chang, "Tournament selection, niching and the preservation of diversity," in *IlliGAL Report No. 91011*, 1991.

[82] E. Alba and B. Dorronsoro, "Cellular genetic algorithms," in *Operations Research / Computer Science Interfaces.* Springer, 2008.

[83] M. Kirley, "A cellular genetic algorithm with disturbances: Optimization using dynamic spatial interactions," *Journal of Heuristics, Kluwer Academic Publishers*, vol. 8, pp. 321–342, 2002.

[84] D. Simoncini, P. Collard, S. Verel, and M. Clergue, “On the influence of selection operators on performances in cellular genetic algorithms,” in *Proceedings of IEEE Congress on Evolutionary Computation (CEC’07).* IEEE, 2007, pp. 4706 –4713.

[85] U. Kohlmorgen, H. Schmeck, and K. Haase, “Experiences with fine-grained parallel genetic algorithms,” *Annals of Operations Research*, vol. 90, pp. 203–219, 1996.

[86] T. Murata and K. Takada, “Performance evaluation of a distributed genetic algorithm with cellular structures on function optimization problems,” in *Proceedings of 8th International Conference on Knowledge-Based Intelligent Information and Engineering Systems.* Springer-Verlag, 2004, pp. 1128 – 1135.

[87] D. Simoncini, P. Collard, S. Verel, and M. Clergue, “From cells to islands: An unified model of cellular parallel genetic algorithms,” in *Proceedings of the 2006 International Conference on Cellular Automata.* Springer-Verlag, 2006, pp. 248 –257.

[88] E. K. Burke, S. Gustafson, and G. Kendall, “Diversity in genetic programming: An analysis of measures and correlation with fitness,” IEEE, Ed., vol. 8. IEEE Transactions on Evolutionary Computation, 2004, pp. 47–62.

[89] A. Morales-Reyes, N. Haridas, A. Erdogan, and T. Arslan, “Adaptive fault tolerant gps attitude determination system,” in *Proceedings of IEEE Aerospace Conference.* IEEE, 2009, pp. 1 – 8. [Online]. Available: ieeexplore.ieee.org/

[90] Y. Xie and Y. Ma, “Design space exploration for 3d integrated circuits.” IEEE, 2008.

[91] R. Yarema, “Fermilab initiatives in 3d integrated circuits and soi design for hep.” ILC VTX Workshop, 2006.

[92] A. Rahman, S. Das, A. Chandrakasan, and R. Reif., “Wiring requirement and three-dimensional integration technology for field programmable gate arrays,” vol. 11. IEEE Transactions on Very large Scale Integration Systems, 2003.

[93] A. Morales-Reyes, A. Erdogan, and T. Arslan, “A distributed cellular ga based architecture for real time gps attitude determination,” in *Proceedings of the 2009*

IEEE Congress on Evolutionary Computation (CEC 2009). IEEE, 2009, pp. 2049 – 2054. [Online]. Available: ieeexplore.ieee.org/

[94] A. Al-Naqi, A. Erdogan, and T. Arslan, "Balancing exploration and exploitation in adaptive three-dimensional cellular genetic algorithm via probabilistic selection operator," in *Proceedings of the NASA/ESA Conference on Adaptive Hardware and Systems.* IEEE, 2010. [Online]. Available: ieeexplore.ieee.org/

[95] ——, "Fault tolerance through automatic cell isolation using three-dimensional cellular genetic algorithms," in *Proceedings of IEEE Congress on Evolutionary Computation.* IEEE, 2010. [Online]. Available: ieeexplore.ieee.org/

[96] J. M. Whitacre, R. A. Sarker, and Q. T. Pham, "The self-organization of interaction networks for nature-inspired optimization," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 2, pp. 220–230, 2008.

[97] B. Dorronsoro and E. Alba, "A simple cellular genetic algorithm for continous optimization," in *Proceedings of 2006 IEEE Congress on Evolutionary Computation.* IEEE, 2006.

[98] D. Ortiz-Boyer, C. Hervs-Martnez, and N. Garca-Pedrajas. (2008) Benchmark problems @ONLINE. [Online]. Available: http://www.cs.cmu.edu/afs/cs/project/jair/pub/volume24/ortizboyer05a-html/node6.html

[99] H. Bersini, M. Dorigo, S. Langerman, G. Geront, and L. Gambardella, "Results of the first international contest on evolutionary optimisation (1st iceo)," in *Proceedings of IEEE International Conference on Evolutionary Computation.* IEEE, 1996, pp. 611–615.

[100] S. Khuri, T. Bäck, and J. Heitkötter, "An evolutionary approach to combinatorial optimization problems," in *Proceedings of the 1994 Computer Science Conference (CSC'94).* ACM Press, 1994, p. (to appear). [Online]. Available: citeseer.ist.psu.edu/khuri94evolutionary.html

[101] K. D. Jong, M. Potter, and W. Spears, "Using problem generators to explore the effects of epistasis," in *Proceedings of 7th International Conference on Genetic Algorithms.* Morgan Kaufmann, 1997, pp. 338–345.