An Empirical Investigation of Optimisation in Dynamic Environments Using the Cellular Genetic Algorithm

Michael Kirley

Environmental and Information Sciences
Charles Sturt University
PO Box 789 Albury, NSW, 2640
Australia
mkirley@csu.edu.au

David G. Green

Environmental and Information Sciences
Charles Sturt University
PO Box 789 Albury, NSW, 2640
Australia
dgreen@csu.edu.au

Abstract

Many real-world optimisation problems are dynamic. For such problems the goal is to track the progression of optimal solutions across the fluctuating fitness landscape rather than to find an exceptionally good solution for a static instance of the problem. Here we present a novel approach for creating robust solutions for non-stationary problems using the Cellular Genetic Algorithm (CGA). The CGA maps the evolving population of solutions onto a pseudo landscape. Intermediate disturbances (disasters) are introduced that break down the connectivity in the pseudo landscape, leading to isolated subpopulations. The dynamic spatial structure of the CGA helps to maintain population diversity. We investigate the performance of the algorithm using a proposed benchmark problem. Simulation results indicate that the CGA is able to respond and adapt effectively to the dynamic environment.

1 INTRODUCTION

For many real-world optimisation problems the environment fluctuates, leading to dramatic changes in the quality of individual solutions. In job scheduling, for instance, the problem is continually changing as jobs are completed and new jobs are added. To address such problems, we require optimisation methods that are capable of continually adapting the population of solutions to a changing environment.

Most optimisation algorithms attempt to find an optimal solution with respect to a specific, fixed fitness measure. In the case of evolutionary algorithms (EA) a great deal of effort has gone into designing efficient representation schemes and genetic operators so as to produce rapid convergence to a good solution. The rapid decrease in diversity of the population results in a highly fit, but homogeneous population, which does not allow the algorithm to perform well in changing environments.

In recent years EAs have been applied to a range of dynamic optimisation problems with varying degrees of success (For example Branke (1999), Grefenstette (1999), Lewis *et al.*, (1998), Mori *et al.* (1997)). Typically, additional heuristics are incorporated into the algorithm to help deal with the non-stationary environment. The previous approaches can be loosely classified into three categories: 1. self-adaptation mechanisms, 2. maintenance of diversity, and 3. memory techniques (Branke, 1999a). The effectiveness of the enhanced EAs depends upon the manner in which the environment changes (De Jong, 1999).

The major challenges when using EAs to solve dynamic optimisation problems are to maintain diversity (or generate diversity) in the population and to evolve robust solutions that are able to track the optima. Ideally we want an adaptive algorithm that responds in an appropriate way every time a change in the environment is detected. In this paper we present a novel approach for creating robust solutions for non-stationary problems using the Cellular Genetic Algorithm (CGA) (Kirley *et al.*, 1998). To enhance the standard genetic algorithm we draw upon ideas from population dynamics and landscape ecology, thereby producing populations and gene pools that are highly adaptive. The rationale behind this approach is that by mimicking nature more closely we will be able to produce a range of good solutions for a given problem in a changing fitness landscape (see section 3).

The structure of the remaining sections of the paper are as follows. In section 2 a brief description of previous work using EAs for dynamic optimisation problems is given. This is followed by a detailed description of the CGA model. In section 4 simulation results are presented based on a benchmark problem proposed by Grefenstette (1999). We conclude with a discussion of the implications of the results obtained and future research questions.

2 DYNAMIC ENVIRONMENTS

2.1 CHANGING LANDSCAPE

In natural environments the manner in which individuals interact is constantly changing. The changes may be random or deterministic. In the case of non-stationary optimisation problems, if we can identify the "dynamics of change" we will be in a better position to solve the problem. Consider the situation where the fitness landscape changes in an arbitrary manner or where the rate of change of the environment increases. In these scenarios we would expect any EA to struggle.

When designing EAs for non-stationary problems it is important to examine the dynamics of the environment relative to the rate of evolution. In this study we will investigate the performance of the CGA in three different changing environments based on the classification scheme proposed by De Jong (1999):

1. Oscillating – the landscape cycles between a small number of states. (De Jong - type 3).
2. Drifting – the landscape changes slowly over time. The peaks move a small distance each generation (De Jong - type 1).
3. Abrupt – the landscape undergoes major changes in a non-deterministic manner. The changes represent "cataclysmic events" (De Jong - type 4).

When a change in the environment has been detected, relevant information about the change (the new fitness function) must be passed to the algorithm.

2.2 PREVIOUS EVOLUTIONARY APPROACHES

In some circumstances it is possible to deal with dynamic optimisation problems as a series of static problems. For example, whenever a change in the environment is detected, the EA could be restarted on the new problem. However, this approach does not use information (partial solutions) gained from the earlier problems.

A number of researchers have proposed different EA models for dynamic optimisation problems. Cobb (1990) introduced the notion of hypermutation. In this model when a change in the environment is detected the mutation rate increases significantly for a fixed number of generations. Grefenstette (1999) goes on to investigate further the effects of evolvable mutations for fluctuating environments. In related work, Angeline (1997) has used self-adaptive parameters to track moving optima.

The inclusion of "memory" or redundant genetic material has been proposed by a number of researchers. Here memory may be in the form of diploid (multiploid) genes with a corresponding dominance mechanism (Goldberg and Smith, 1987, Lewis *et al.*, 1998). Dasgupta and McGregor (1994) made use of a hierarchical tree-based chromosome that constituted a form of long term memory.

An alternative memory mechanism is the use of an external pool of solutions, where selected individuals (usually highly fit) are stored and reintroduced into the evolving population in later generations. In the Thermodynamical Genetic Algorithm (Mori *et al.*, 1998), the best individual from each generation is stored in memory. The memory size is fixed and thus another individual is deleted from the memory depending on its age and the genetic diversity in the memory pool. Branke (1999b) compares a number of replacement strategies for inserting new individuals into memory in a similar study.

The results reported in the papers listed above indicate that the effectiveness of the enhanced EAs depends upon the manner in which the environment changes. Memory based techniques appear to be better suited to periodically changing environments. The self-adapting and diversity maintaining algorithms are preferable in other environments. In non-stationary landscapes, the trade-off between selection and variation, and the corresponding impacts on population diversity, is a critical performance issue. (For a comprehensive review of EA based approaches for dynamic optimization problems see Branke (1999a)).

3 CELLULAR GENETIC ALGORITHM

The CGA was introduced by Kirley *et al.* (1998). A discussion of the biological basis of the algorithm appears in Green and Kirley (2000). In the CGA the evolving population is mapped on to a two dimensional toroidal grid; a pseudo-landscape. Computationally the CGA is a fine-grained parallel genetic algorithm, but with certain biologically inspired modifications. Whitley (1993) introduced the term "cellular genetic algorithm" for this sort of model. We add the potential to dynamically create isolated sub-populations by including occasional "disasters", which clear patches of the cells in the pseudo-landscape.

The CGA extends the basic genetic algorithm in the following ways:

1. It maps the population of solutions onto a toroidal grid. This grid serves as a pseudo landscape in which each cell represents a portion of the landscape. It should be emphasised that this is entirely unrelated to so-called fitness landscapes. Instead it is meant to resemble real landscapes with the solutions being akin to (say) plants spread out on the landscape.
2. In each cell, the genes comprise the state of the cell, and breeding is confined to crossover between cells within the same neighbourhood.
3. The cells in the CGA grid are considered to be directly "connected" if one belongs to the neighbourhood of the other. The usual definition of connectivity from graph theory and topology applies: two cells are connected if there is a sequence of pairs of directly connected cells between them.

4. To the normal GA processes of mutation and culling, the CGA adds spatial processes:

   4.1. Some cells may be designated as "inactive" (ie. uninhabitable), which means that they never contain members of the population of solutions and take no part in crossover and other processes.

   4.2. In the course of a run, some cells may be temporarily empty, which mean that no member of the population occupies them.

   4.3. Intermittent "disasters" clear patches of cells, leaving them temporarily empty. Disasters are characterized by two parameters: their *size* (the number of cells cleared) and their *frequency*.

   4.4. Empty cells can be reinvaded by "seeding" from live cells within the neighbourhood.

The CGA model exploits changes in landscape connectivity to make the population and gene pools highly adaptive. The introduction of disasters, which fragments the population, moves the algorithm into a different "phase" where variation rather than selection dominates. As the nearest neighbours slowly reclaim the cleared cells, the algorithms gradually shifts back into its original phase (Figure 1). When individuals from the fragmented population come into contact and mate, fitter hybrids often appear.

The restricted mating scheme, combined with the dynamic topology employed in the CGA, helps to maintain diversity in the evolving population (Kirley *et al.*, 1998). However, when we are confronted with fluctuating fitness landscapes, it is important to track near optimal solutions as well. For non-stationary problems the following modifications have been proposed when the environment changes:

1. Full connectivity in the pseudo landscape is restored. That is, all empty cells are reloaded (initialised).
2. An elite migration policy is implemented. The best individual from each static instance of the problem is stored in an external immigration (memory) pool. When a change in environment is detected, the elite migrants are reintroduced into the population at the sites of previous disasters.

4 EXPERIMENTS

4.1 TEST PROBLEM

Recently, there has been a push to develop new benchmark problem generators for dynamic environments (Branke, 1999, Grefenstette, 1999, Morrison and De Jong, 1999). Branke suggests that problems such as the time-varying knapsack problem used by Lewis *et al.*, (1998), and scheduling problems used by Louis and Xu (1996)

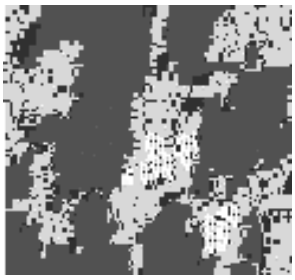

Figure 1: Patchy sub populations in the CGA. Black indicates disaster zones, light colours indicate fitter individuals. Isolated patches have formed as a result of the accumulation of "disasters" in the pseudo landscape.

are not suitable as benchmarks. The periodically changing environments and the implicit assumptions built into the problems do not allow enough insight into the working of the optimisation algorithm.

For our investigation we have used the dynamic fitness landscape proposed by Grefenstette (1999). This artificial landscape is made up of number of peaks of varying heights, which can be changed independently using runtime parameters. Individuals in the CGA population are interpreted as points in *n* dimensional space:

$$x = \{ x_1, x_2, \ldots x_n \}$$

The fitness landscape is defined as follows:

$$f(x) = \max g_i(x)$$

The landscape is specified as a set $\{ g_i \}$. Each g_i is a component in the landscape consisting of a single time varying *n* dimensional Guassian peak (see Figure 2).

The fitness contribution to point x from g_i is defined as:

$$g_i(x) = A_i(t)\exp(-d(x,c_i(t))^2/(2\sigma_i^2(t)))$$

where:

$A_i(t)$	Amplitude – is the fitness contribution obtained by an individual located at the centre of the peak.
$c_i(t)$	Centre – specifies the coordinates associated with the maximum value of the peak at time t.
$\sigma_i(t)$	Width – specifies how the fitness contribution from this peak decreases as a function of the distance from the centre of the peak.
$d(x,c_i(t))$	Euclidean distance between x and the peaks centre.

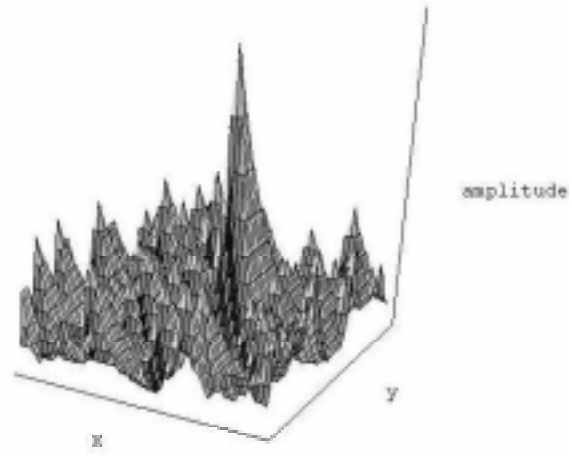

Figure 2: Fitness Landscape. A 3D plot of a static view of the dynamic landscape. When the environment changes, all peaks move in a random direction based on the run-time parameters. The maximum peak amplitude = 100.

This problem is ideal because the rate of change of the environment can be controlled, the landscape is rugged in the sense that fitness is defined as the maximum contribution of all peaks and, finally, the problem is scalable.

4.2 MODEL PARAMETERS

To create the non-stationary landscapes we adopt the same parameter settings for the test problem as Grefenstette. The initial landscape consists of 256 peaks uniformly distributed over the 2 dimensional region bounded by (0,0) and (100,100). All peak amplitudes were initialised uniformly from the interval [10,50]. One randomly select peak is then assigned the optimal amplitude of 100. All peak widths were set to 4, ie. the peak's fitness contribution drops to about 50% of its amplitude at a distance of 4 from its centre. The dynamic landscape is implemented by moving all peaks in randomly selected directions over time. It is important to note that the amplitudes are kept constant in all environments.

To explore the effectiveness of the CGA we ran experiments using the three different dynamic environments identified in section 2. The motion of individual peaks was controlled by two run-time parameters:

1. *drift rate* - the distance each peak moves drawn from the uniform interval [0,50].
2. *punctuation rate* - how often the peaks move (the number of generations before the next change).

For the drifting environment the *punctuation rate* = 1 (every generation). In the abrupt environment the *punctuation rate* = 20. The magnitude of the change was equivalent to 50 generations of gradual change associated with the drifting environment. For the oscillating environment the landscape cycles between two different states created using the abrupt method. Once again the *punctuation rate* = 20.

The CGA model parameters are based on previous experiments in static optimisation (Kirley *et al.*, 1998). A 40 bit binary chromosome (2×20 bits strings, one for each dimension) was used to encode solutions. The grid size was set to 15, ie. a population of 225 individuals. A fitness nearest neighbour strategy was used for mate selection. In this strategy the fitness values of each of the eight adjacent cells are sampled. In a form of tournament selection, the fittest individual from the local area is selected as the mating partner. The crossover rate and mutation rates were set to 0.6 and 0.05 respectively.

For each of the non-stationary landscapes four different EA configurations were examined; a standard genetic algorithm (GA) as described by Michalewicz (1996) and three different connectivity parameter settings for the CGA. The specific parameter settings were:

1. GA (standard genetic algorithm)
   - population size = 225
   - crossover rate = 0.6 (one point crossover)
   - mutation rate = 0.05
2. FG (fine-grain model)
   - local neighbourhood size = 1
3. Disaster (FG + intermediate rate of disturbances)
   - disaster size (zone radius) = 5 cells
   - disaster frequency = 0.3
4. Reload (disasters + memory)
   - disaster size (zone radius) = 5 cells
   - disaster frequency = 0.3
   - random reloading of empty cells
   - elite migration

4.3 RESULTS

The CGA and standard GA models were written in Java and implemented on a Sun Ultra Sparc 1. All configurations were executed 50 times for a maximum of 200 generations. The optimum value in each landscape configuration is 100 (Figure 2).

To compare the performance of each configuration under investigation we use the *current-best metric* described by Grefenstette (1999). The *current-best metric* is the average value of the best fitness value in the current population. Figures 3 displays results for each of the dynamic landscapes examined. In Figure 4 we compare two additional time-averaged measures, online and offline

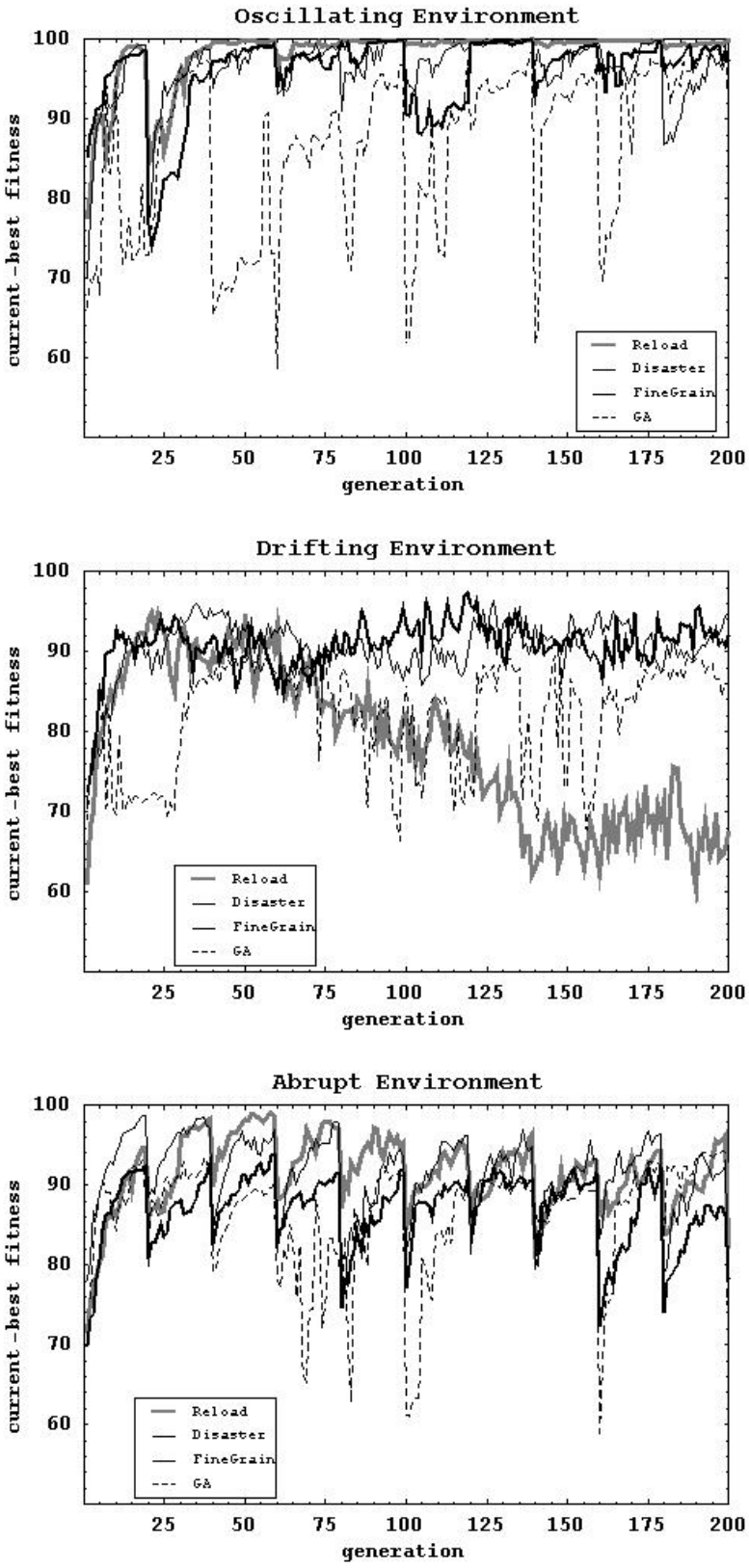

Figure 3: Current-best fitness *vs* generation for the each of the dynamic environments.

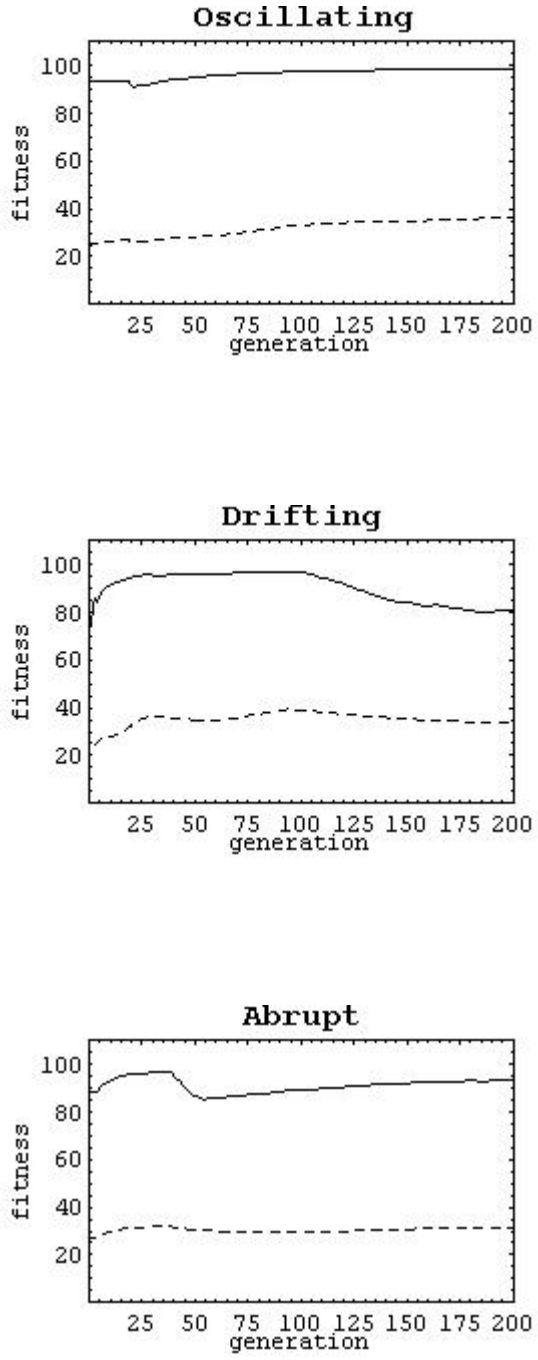

Figure 6: Time-average performance measures: Online performance (dashed line) and Offline performance (solid) for the reload configuration for each of the test landscapes.

performance for the reload parameter settings. Offline performance is a running average of the best solutions at each timestep. Online performance is simply the mean fitness for all individuals generated in a given run.

5 DISCUSSION

The CGA can be seen as a hybrid between a fine-grained and a coarse-grained parallel GA. External stimuli (disasters) alter the density of occupied cells in the grid leading to abrupt changes in connectivity. These changes generate population responses very different from the slow, neo-Darwinian dynamics of standard GAs. The resulting "patchy populations" that are formed maintain some degree of independence allowing the exploration of different regions of the search space. Consequently the algorithm is able to explore novel solutions while simultaneously refining optimal ones.

For non-stationary problems our objective is to track the progression of good solutions across the fluctuating fitness landscape. In this study, we compare the performance of three different configurations of the CGA and a canonical genetic algorithm. The CGA model offers the advantage of providing a flexible, adaptive population that is able to respond to a changing environment.

Figure 3 provides a summary of the CGA performance in each of the dynamic landscapes. In the oscillating environment there is a significant difference between the configurations examined. In the standard GA mode the current best individual value fluctuates significantly throughout the run. As the number of generations approaches 200 the fitness value slowly increases towards the known optimum. In the fine-grain and disaster modes the algorithm was able to track near optimal solutions, with an associated time delay. When the elite migrants were reintroduced into the population (reload mode), the algorithm was able to move the population towards a new global optimum almost instantaneously.

For the drifting environment (Figure 3), an examination of the standard GA plot reveals a large variation in the current best value. In various stages (for example, generations 20-30, 30-60) the algorithm appears to be trapped in local optima. The performance of the fine-grain and disaster configurations was very similar. Here the optimum value in each generation fluctuated around the mid 90s mark. The plot for the reload mode provides an interesting result. In this configuration, when a change in environment is detected, any vacant site in the pseudo-landscape is re-initialised. Elite migrants are also re-introduced into the population. The poor performance depicted in this plot can be explained by the rate of change of the environment. The population is unable to adapt fully to the new problem before the environment changes again. The influx of elite migrants each time the

landscape changes, has the negative effect of steering the search into areas already examined. For the given population size, after generation 100, more than half the population in each generation is made of up individuals reloaded from memory.

The abrupt environment is the final plot in Figure 3. In this scenario the problem changed by a significant amount every 20 generations. The CGA was able to track a near optimal value. The fine-grain and disaster modes produced similar results. The spatial distribution of the CGA population provides the necessary basis to maintain diversity of solutions. In the reload mode there appears to be slightly less variation in the average fitness values. However, the difference is not significant. In the standard GA plot we see that changes to the fitness landscape have dramatic effects on the current best value at times (for example, generations 75-100, 160). As was the case in each of the other dynamic environments, the standard GA takes a longer time to adapt to the changing environment.

To further explore the implications of the modifications included in the CGA for dynamic environments, we focus our attention on time-averaged performance measures. In Figure 4 online and offline measures for a typical trial are plotted for the reload mode. The online plots are very similar for all dynamic environments, starting at approx 25 and increasing to approx 35. In the earlier generations the introduction of disasters tends to decrease the online performance as expected. Similar trends are evident in the offline performance. Variations in the offline performance may be attributed to the rate of change of the environment.

6 SUMMARY AND FUTURE WORK

The experimental results presented in this paper indicate that the CGA is able to respond and adapt effectively to dynamic environments. In the case where the environment oscillates between states, the use of elite migrants improved the overall performance. In the other non-stationary environments considered, the spatial population structure helps to maintain diversity in the population, and consequently solution quality. It was found that the reload option actually inhibits the search for optimal solutions if the rate of change of the environment is too great. In all dynamic environments considered the performance of the standard GA, when tracking the optimal value, was inferior to the CGA. The dynamic communication topology of the CGA provides a biologically based approach for solving optimisation problems in dynamic environments.

In future work we will examine the performance of the CGA on other proposed benchmarks and real-world problems. An interesting research question is to analyse the effectiveness of using the patchy sub populations to evolve solutions for static instances of the dynamic problem at the same time.

References

P.J. Angeline (1997). Tracking extrema in dynamic environments. In *Sixth International Conference on Evolutionary Programming.* pp.335-345. Springer-Verlag.

J. Branke (1999a). Evolutionary Approaches to dynamic optimization problems - A Survey. In A. Wu (ed.) *GECCO Workshop on Evolutionary Algorithms for Dynamic Optimization Problems.* pp. 134-137.

J. Branke (1999b). Memory-Enhanced Evolutionary Algorithms for Changing Optimization Problems. In *Proceedings of the Congress on Evolutionary Computation (CEC'99), IEEE*, vol.3 pp. 1875-1882.

H. Cobb (1990). *An investigation into the use of hypermutation as an adaptive operator in genetic algorithms having continuous time-dependent nonstationary environments*. Technical Report AIC-90-001. Naval Research Laboratory, USA.

D. Dasgupta and D.R. McGregor (1992). Nonstationary function optimization using the structured genetic algorithm. In B. Manderick (ed) *Parallel Problem Solving from Nature - PPSN II.* pp 145-154. Elsevier.

K. De Jong (1999). Evolving in a Changing World. In Z. Ras and A. Skowron (eds) *Foundation of Intelligent Systems. Lecture Notes in Artificial Intelligence 1609* pp 513-519. Springer.

D. Goldberg and R. Smith (1987). Nonstationary function optimization using genetic dominance and diploidy. *In Proceedings of the Second International Conference on Genetic Algorithms.* pp. 59-68. Morgan Kaufmann.

D.G. Green and M.G. Kirley (1999). Adaptation, diversity and spatial patterns. *Knowledge-Based Intelligent Engineering Systems 4 (2).*

J. Grefenstette (1999). Evolvability in Dynamic Fitness Landscapes: A Genetic Algorithm Approach. *In Proceedings of the Congress on Evolutionary Computation (CEC'99), IEEE*, vol.3 pp. 1875-1882.

M. Kirley, X. Li, and D.G. Green. (1998). Investigation of a cellular genetic algorithm that mimics landscape ecology. In X.Yao *et al.* (eds), *Simulated Evolution and Learning SEAL98, Lecture Notes in Artificial Intelligence 1585*. pp. 90-97. Springer.

J. Lewis, E. Hart and G. Ritchie (1998). A Comparison of Dominance Mechanisms and Simple Mutation on Non-Stationary Problems. In A. Eiben *et al.* (eds) *Parallel Problem Solving from Nature - PPSN V. Lecture Notes in Computer Science 1498.* pp 119-128. Springer.

S. J. Louis and Z. Xu (1996). Genetic algorithms for open-shop scheduling and re-scheduling. In M. Cohen *et al.* (eds) *ISCA Eleventh International Conference on Computers and Their Applications*. pp 99-102.

Z. Michalewicz (1996). *Genetic Algorithms + Data Structures = Evolutionary Programming - Third Revised Edition.* Springer.

N. Mori, H. Kita and Y. Nishikawa (1997). Adaptation to changing environments by means of the thermodynamical genetic algorithm. In A. Eiben *et al.* (eds) *Parallel Problem Solving from Nature - PPSN V. Lecture Notes in Computer Science 1498.* pp 149-158. Springer.

R.W. Morrison and K.A. De Jong (1999). A test problem generator for non-stationary environments. *In Proceedings of the Congress on Evolutionary Computation (CEC'99),* pp. 2047-2053.IEEE.

D. Whitley (1993). Cellular genetic algorithms. In S. Forest (ed). *Proceedings of the 5th Int. Conference on Genetic Algorithms*. pp.658-662. Morgan Kauffmann.