Genetic Algorithms: Combining Evolutionary and 'Non'-Evolutionary Methods in Tracking Dynamic Global Optima

Simon M Garrett
Dept. Computer Science
University of Wales, Aberystwyth
Wales, SY23 3PG. UK
smg@aber.ac.uk

Joanne H Walker
Dept. Computer Science
University of Wales, Aberystwyth
Wales, SY23 3PG. UK
jhw94@aber.ac.uk

Abstract

The ability to track dynamic functional optima is important in many practical tasks. Recent research in this area has concentrated on modifying evolutionary algorithms (EAs) by triggering changes in control parameters, ensuring population diversity, or remembering past solutions. A set of results are presented that favourably compare hill climbing with a genetic algorithm, and reasons for the results are suggested. A method is then introduced, Evolutionary Random Search (ERS), that combines crossover and hill climbing mutation in a novel manner. It is assessed against the GA and hill climbing tests, and the encouraging results are discussed.

1 Introduction

There is a growing literature concerning the finding and tracking of *dynamically changing optima*, by evolutionary computation, e.g. Branke (1999a); Grefenstette (1999); Oppacher and Wineberg (2000); Eggermont et al. (2001); Ursem (2000). Almost all of these methods take the form of modified genetic algorithms (GAs), although some have compared GAs and Evolution Strategies (ES), e.g. De Jong (1999). None could be found that used hill climbing in dynamic optimisation. This paper provides an updated summary of the literature in this field and investigates the value of hill climbing versus GAs in tracking optima.

Four optimisation methods — two types of GA and two types of Random Mutation Hill Climbing (RMHC) — were applied to optimisation tasks of different types, and levels of complexity. The results agree with those of Mitchell et al. (1993) who found that RMHC was more effective than typical GA approaches. The work presented here suggest that this is also true when following *dynamically* changing optima.

Continuing from the comparative results, suggestions are made about how a new hybrid evolutionary-hill climbing method might be obtained, by combining elements from GAs and RMHC. Three versions of the new method, Evolutionary Random Search (ERS), are tested on the optimisation tasks. Conclusions are drawn from the results and suggestions are made about the relationship between ERS, RMHC and GAs.

2 Background

This section reviews previous work on: (i) comparisons of EAs and hill climbing, and (ii) evolutionary techniques for dynamic optimisation.

2.1 EAs and Hill Climbing

In 1993, Mitchell et al. (1993) asked the question, "When will a genetic algorithms outperform hill climbing?", in work that set out to test Holland's Building Block Hypothesis (1975). To this end an experiment was devised that, it was thought, would lay out a simple *Royal Road* for a GA to follow, from small schemas through to the optimal string, and compared it to three hill climbing methods. Instead it was found that one hill climbing algorithm, RMHC, outperformed the GA by an order of magnitude. RMHC was defined as follows, "In RMHC, a string is chosen at random and its fitness is evaluated. The string is then mutated at a randomly chosen single locus, and the new fitness is evaluated. If the mutation leads to an equal or higher fitness, the new string replaces the old string[1]. This procedure is iterated until the optimum has been found, or a maximum number of function evaluations has been performed", on a population of individuals.

[1] Note that this mutation operator is *not* destructive.

The main area of weakness in the GA was identified as a property called *hitchhiking*, that occurs when an unfit gene near to a fit schemas is spread through the population along with the fit schema. One of the reasons RMHC outperformed the GA was because the GA was doing wasteful work removing these hitchhikers, whereas RMHC was not.

Lang (1995) suggested that hill climbing improved on results in Koza (1992), beginning a series of claims and counter claims, involving Koza's informal rebuttal at the ML-95 conference and hearty discussions on the internet (Lang et al., 1995). Both Lang and Koza seemed to overlook Lang calling his hill climbing method 'RMHC' in their internet exchanges. According to Lang (1995), and the definition given above, the method used was *not* RMHC since it involved crossover, and apparently did not create new random solutions as mutations of previous solutions.

True RMHC can be considered to be a form of parallel evolution, akin to multiple, isolated (1+1)-ES populations, in which the population's single member competes only with its offspring for survival. However, in RMHC the degree of mutation is not usually under the control of strategy parameters.

2.2 EAs and the Dynamic Optimisation Problem

The motivation for our work is, however, quite different from Mitchell et al. (1993). It springs from a need to optimise robot behaviours in a changing environment, see Walker and Wilson (2002), and hence relates to a real-world problem in evolutionary dynamic optimisation.

Angeline (1995) and Branke (1999a) have summarised much of the work in this field, to which the reader is referred for details. However, relevant aspects of their papers, and additional, more recent work, are presented below. Branke usefully categorises this work as exhibiting one of the three following characteristics:

- Triggered change to the EA's operation when a change in optima is encountered.
- Continual diversity so the EA's population can always respond to changes in optima.
- Remembering previous solutions (e.g. previous global optima) to be reused later.

All three approaches aim to mitigate the lack of diversity in a standard, optimised EA population. When diversity is lacking, and the optima change, most (if not all) of the EA's search points can remain located around the old global extreme, and thus they can easily be trapped in new local optima near that point.

2.2.1 Triggered Change

The first approach is that of *triggered change*, an early example of which is Cobb's work in *triggered hypermutation* (Cobb, 1990), improved by Cobb and Grefenstette (1993).

Grefenstette and Cobb's approach triggered a large increase in mutation, when the environment changed, to increase the diversity of the population. The mutated individuals were able to search areas of the fitness space away from the old point of convergence and, by means of the crossover operator, the population could spread throughout the new fitness space, before converging again, at or near the new global extreme. The process can repeat indefinitely. Grefenstette has recently provided a comparison of various types of mutation and hypermutation models (Grefenstette, 1999).

Grefenstette pointed out that Cobb's triggered hypermutation fails in some types of dynamic environments (Grefenstette, 1992). Firstly, if the environment changes significantly and the new optima are not close enough to previous ones, hypermutation may not introduce enough diversity to overcome this. Secondly, if the fitness space changes only by *adding* new optima, then hypermutation would not be triggered at all. The solution involved a proportion of the population being continually replaced by random individuals, an approach called the *Random Immigrants GA*. In a comparison between the performance of a standard GA, triggered hypermutation and random immigrants algorithms, Cobb and Grefenstette (1993) found that in dynamic environments with large scale changes, random immigrants performed best, it also caused less disruption in stable environments.

The Random Immigrants method was further modified (Grefenstette, 1999), where an attempt was made to control the mutation rate genetically, in a similar manner to an ES. Grefenstette used this in a number of adaptive mutation rate tests, and found that, although techniques using some hypermutation gave better results than those that did not (for both gradually and suddenly changing environments), altering the mutation rate dynamically was not as effective.

2.2.2 Continual Diversity

Grefenstette's work using random immigrants might be thought of as a move towards *continually ensuring* high diversity, in place of regaining population diversity when change is detected. This section discusses other means of maintaining continual diversity.

Ghosh et al. (1998) implemented an aging population of individuals because dynamic optimisation was considered to be, "...optimizing a series of time-dependent optima." This approach added to the Steady State GA (SSGA) such that each individual was given an age, used in calculating its fitness (along with performance metrics), with middle-aged individuals gaining the most fitness. It was found that his form of GA improved on the performance of the SSGA in both stationary and non-stationary environments. In dynamic environments the new version outperformed the SSGA for small environmental changes, and was particularly impressive for large scale changes.

Ursem's comprehensive work (Ursem, 2000) set out a relatively complex algorithm, which views the GA's population as a number of subpopulations that may be climbing different peaks in the fitness landscape. These subpopulations were identified and maintained during subsequent generations, for instance by chosing crossover mates from within the same subpopulation. It was found to outperform the sharing GA (Goldberg and Richardson, 1987).

Another project using subpopulations took a simpler approach (Oppacher and Wineberg, 2000), a main "core" population, and numerous "colonies" around it that explored different parts of the fitness landscape. These colonies were forced away from the core population's fitness space, and performed hill climbing to avoid unnecessary exploration of poor parts of the landscape. Crossover occurred within colonies and the core only. Good colony members could periodically migrate to the core, to provide it with increased diversity and to allow it to adapt quickly when the environment changed. It was found to outperform a standard GA in dynamic environments.

2.2.3 Remembering Solutions

If a new optimisation problem is similar to a previous solution, it might be more efficient to remember the old solution than to regenerate it. Again Branke makes a useful distinction here between two types of approach: firstly, *implicit memory* often using multiploidy, e.g. Dasgupta and McGregor (1992) and Lewis et al. (1998), although neither appears to be particularly robust; secondly, *explicit memory* where solutions are specifically stored for later reuse. With Ramsey, Grefenstette has addressed this issue too (Ramsey and Grefenstette, 1993), but this does assume that the environment can be measured, so that the relevant solution can be retrieved.

The method described in (Louis and Xu, 1996; Louis and Johnson, 1997) sampled and remembered the best member of the population at regular intervals. It was found that seeding the next run of the GA with 5-10% of the remembered individuals gave improved results. However the method appeared to be fragile to a higher percentage of seeding, and to large changes in the fitness space.

Eggermont et al. (2001) have reported a GA with a case based memory of past successes, which the GA accessed when the environment changed. After each generation, the best individual was added to the memory, then when the fitness deteriorated, e.g. due to environmental change, the individuals in memory were re-evaluated and the best in the current environment was re-introduced into the population. It was found that this case-based addition to the GA improved the performance in a dynamic environment.

Branke (1999b) himself describes a method that defined two populations: a memory population to remember previous, good solutions, to maintain a minimum degree of quality; and a population to search constantly for new peaks, submitting its best efforts to the memory population. The search population was reinitialised after every change in the fitness space.

3 Aims and Objectives

In light of the research, just reviewed, the aim of this work is to answer two questions: "Can RMHC and GAs be combined to produce a method that ensures a continually diverse population?" and, "How will such a method perform, relative to RMHC, and why?" The objectives to obtain these aims are two-fold:

Firstly, to compare the performance of GA and RMHC methods in a dynamically changing, multidimensional environment, under a variety of conditions. The different conditions should test different aspects of the methods and highlight their benefits.

Secondly, to use this information, and previous work in evolutionary dynamic optimisation, to begin to generalise a new method of optima-tracking. This method should have a performance which approaches or exceeds RMHC, and which improves on standard GA performance.

4 Methodology

4.1 Optimisation tasks

The optimisation tasks to be solved were 2-, 5- and 10-dimensional versions of Equation 1. Increasing the number of dimensions, from 2, to 5 to 10, made the optimisation task more difficult, since this reduced the density of search points in the fitness space. In all cases the space searched by the methods below was limited to an $0 \le x_i < 10$ square/hypercube. The standard benchmark equation to be optimised was,

$$f(\mathbf{x}) = \frac{sin(5(\sum_{i=1}^{N}(x_i - xf_i)^2)^{1/2})}{5(\sum_{i=1}^{N}(x_i - xf_i)^2)^{1/2}} \tag{1}$$

for $N \in \{2, 5, 10\}$, and where f defines the fitness at a point in space represented by the genes of an individual[2], with $f \to 1$ as the Euclidean distance from the origin to $\mathbf{x} \to 0$.

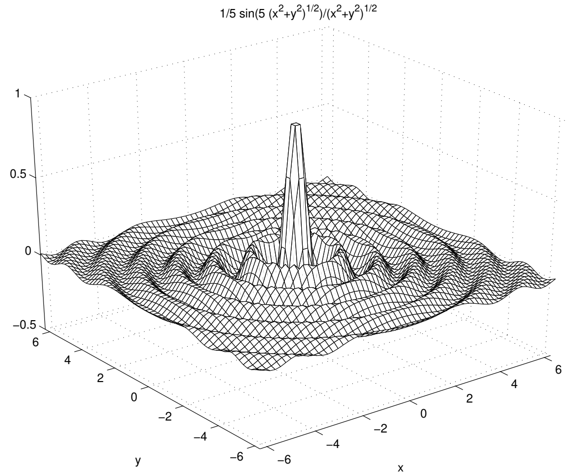

Figure 1: A 2-D graph of Equation 1, here the global maximum is at (0,0)

To implement the dynamics required, the global optimum (in this case a maximum) was relocated by adding an offset, xf_i, to each x_i. A change occurred every 1000 generations, twenty times. The location of the maximum changed in two ways:

- Randomly within the space [(0,0),(10,10)). This tested the ability of the algorithms to find optima that were often well-separated in the fitness space, with the complication of intervening local optima. Methods with poor continual population diversity should not perform well.
- Incrementally from (0,0) to (2,2), in 0.1 increments. This tested the ability of the algorithm to track optima that were relatively close in the fitness space. In some cases, parts of the maximal peak were outside the search space. This made the search task slightly harder than if the maximum were always totally with the space.

[2] Refers to both GA and RMHC population elements

4.2 Comparative Tests: the GA and RMHC Methods

The methods used were:

GA An unremarkable implementation, using two-adversary tournament selection. Crossover swapped a two real-valued genes, at a random locus, with a probability of 0.8. The mutation operator, which set the new value of a gene randomly in the range [0,10), had a probability of mutation of 0.01. The GA's control parameters were adjusted by hand to maximise its performance.

GA with Elitism (GA-E) Implemented as described in the previous point, except for the addition of elitism, which retained the best member from each generation.

Global mutation RMHC (RMHC-G) Both forms of RMHC followed the definition in Mitchell et al. (1993), set out above. The only variation was in the type of mutation. RMHC-G used the GA's mutation operator and set a 'gene' in the range [0,10). Note that there was no need to maximise the performance of either form of RMHC.

Local mutation RMHC (RMHC-L) Identical to RMHC-G, described in the previous point, except that the RMHC-L mutation operator ensured that a new 'gene' value, v' was close to the old value v. Given r, the range of possible values for v, this was achieved by, $v' = v + random() * r/100 - r/200$. Unlike the GA, both forms of RMHC are only able to alter one dimension (gene) before having fitness evaluated. This placed them at a disadvantage to the GA methods, in which crossover can alter several dimensions at once.

All methods used real-valued chromosomes, not bit strings, with a population size of 100, and 1000 generations of continuity. Tests showed that raising these levels by an order of magnitude made very little difference in relative performance of the methods, and lowering the values by an order of magnitude only slightly changed their relative performance. Since 100 generations and populations of 10 make the results less repeatable, the values above were preferred and the optimisation tests were run under these conditions for all four methods.

4.3 ERS: Improving on the Standard Methods?

Another approach, introduced here, combined elements of these algorithms to form an *Evolutionary Random Search* (ERS) method. In population terms, an ERS is analogous to removing a number of individuals from a population and shipwrecking them on an island, where they and their offspring compete against each other. During this time they may be considered a subpopulation, as discussed Section 2.2.2, however they do not necessarily represent points near local optima. The ERS *framework* (Figure 2) allows instances of an ERS to vary the number of shipwrecks that occur, and the number and types of offspring. In all cases however, a shipwreck involves some combination of RMHC's non-destructive mutation and GA's crossover. There are t shipwrecks per generation. Three versions of ERS are discussed here. In all three examples, $n = 3$, and P, the population of individuals, had 100 members; but this need not be the case.

The ERS1 method combined GA, RMHC-G and RMHC-L in a deliberately naïve and expensive manner. Its purpose was to show the combined effect of GA, RMHC-L and RMHC-G. During each generation there were $|P|/n$ shipwrecks, covering the whole of the population, bar one individual. This individual could be replaced with a random individual, or with a remembered previously successful individual, to help the recovery of previously seen solutions. For these tests the individual was randomised. Each iteration of the repeating portion of Figure 2 created 12 new individuals. Three of these were global mutants, three were local mutants and 6 were the offspring of the two groups of mutants. Thus the naïvety of ERS1 was due to it being *four times more expensive* than RMHC and the GA.

ERS2, was identical to ERS1 in all but two respects. Firstly, it created only 8 shipwrecks per generation, to ensure nearly the same number of fitness evaluations (96) as the GA and RMHC methods (100), making it possible to compare the effectiveness of ERS2 and RMHC-G. Secondly, it randomly chose the members of its 8 shipwrecks, and the remaining individuals were left unaffected for that generation.

ERS3 tried a different combination of parameters to expore other possibilities of the ERS framework, whist again performing the same number of fitness evaluations as the ERS2, GA and RMHC methods. ERS3 ordered the population by fitness, and split it into three parts of $|P|/3$ individuals (the remaining individual was replaced by a new random individual.) It selected an individual from each of the three parts,

- Repeat t times:
  - Select n members, $\mathbf{s}$, from P.
  - Create m_1 RMHC-G mutants from $\mathbf{s}$.
  - Create m_2 RMHC-L mutants from $\mathbf{s}$.
  - Select members from the local and global mutants.
  - Crossover selected individuals and create children.
  - List originals, mutants and children.
  - Sort list.
  - Place fittest n from list back into P.
- End repeat.
- Replace any remaining individuals in P with random (or remembered) individuals.

Figure 2: The ERS framework (one generation)

called *hi*, *mid* and *lo*, to indicate their relative fitness. Each shipwreck in ERS3 applied crossover, and non-destructive mutation to these three members, in a manner that maximised the chance that the offspring would represent improvements to the optimisation task. To this end, ERS3 dispensed with RMHC-L (i.e. $m_2 = 0$), since the results below will show it provided little in terms of performance, and because RMHC-G can do local optimisation where required, although it takes longer. Both of the RMHC-G mutants were of the *hi* individual because, with no other information available, each was more likely to lead to a fitter individual than mutants of *lo* or *mid*. These two new individuals are called $gm1$ and $gm2$. The remaining individual was created by crossover of *lo* and *mid*, in the hope of occasionally combining two poor solutions into one fit one. This operation also simultaneously altered multiple dimensions, something the mutation operator can not do. One of the two offspring was retained at random to give xi. At this point the fitnesses of $gm1$, $gm2$ and xi were evaluated. The three best individuals, of the six total individuals on the island, were then returned to the population.

5 Results

Figure 3 shows the best and average fitness, for each generation of a GA tracking a 2-dimensional version of the randomly changing function. Since the best and average fitness were closely related in all tests, the average fitness results will be omitted in the following results. The full results set can be found at `http://www.aber.ac.uk/`

smg/WORK/ga-hc-results.htm, or on request. They consist of the best, average and worst fitness for each generation of each experiment, and re-runs of the full set of results to demonstrate repeatability, as well as results for other fitness landscapes.

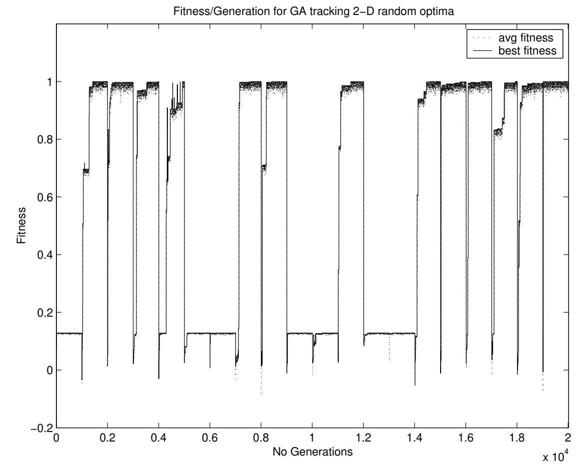

Figure 3: A typical task: the GA optimising a 2-D, randomly changing fitness function

Figure 3 illustrates how a population's best fitness dropped every 1000 generations, when the fitness function's optima were changed. The fitness then often returned to previous levels. At times, however, the GA could not find the global maximum in the allotted number of generations because all population members were caught around a local maximum, and mutation was not occurring often enough to provide search points that were fitter than the local maximum. The average best fitness is used to summarise the success of each of these tests, as shown in Table 1. The standard deviation is also available in the full results.

The test results from Table 1 are presented in Figures 4 and 5. These show the two types of dynamic test, random (left) and incremental (right), for 2-, 5- and 10-dimensional versions of the test function.

5.1 Comparative Test Results

The incremental results for RMHC-L, the GA and the GA-E were fairly dependent on the fitness of the initial population — when high, the optimisation method might track the maximum for a number of generations. However, there were some clear results.

In all cases RMHC-L performed worst, with the exception of the 10-dimensional incremental test, an artefact of the RMHC-L's initial population in the result presented here. The GA and GA-E performed better;

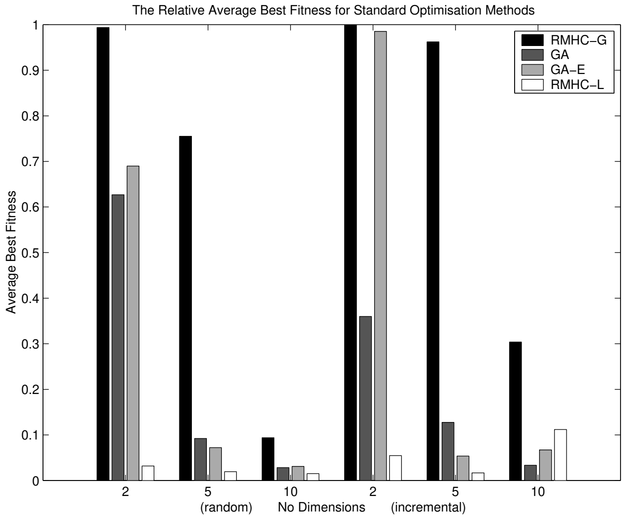

Figure 4: The comparative results: the left-hand group of three plots show the randomly moving optima results and the right-hand group show the incrementally moving optima results, both for 2-, 5- and 10-dimensions

almost equally as well as each other, except for the 2-dimensional incremental test at which GA-E was superior. RMHC-G was by far the most successful method. These results indicate that the results in Mitchell et al. (1993) apply to dynamic optimisation, as well as to static optimisation, under some conditions.

The two types of RMHC performed best and worst, indicating a relationship between the amount of non-destructive mutation and utility. Presumably an ES (without crossover) would perform somewhere Between these two extremes, since Gaussian mutation can result in both large and small changes. This will be tested in future work.

5.2 Evolutionary Random Search (ERS) Test Results

Figure 5 shows that ERS1 was better that RMHC-G in all cases. Since ERS1 performed the same amount of random mutation as RMHC-G, plus extra RMHC-L and crossover operations, this is not surprising. What is perhaps unexpected is that ERS1 does not out-perform RMHC-G by much, except in the 10-dimensional incremental tests. However, when the ERS method is limited to the same amount of processing as other methods (ERS2 and ERS3), the difference in performance between them and RMHC-G was not as clear (although both types were still superior to GA-E, GA and RMHC-L).

Table 1: Data for Figures 4 and 5

	Random			Incremental			
No. Dimensions	2	5	10	2	5	10	Average
ERS1	0.995	0.774	0.152	0.999	0.995	0.977	0.8157
ERS2	0.986	0.224	0.075	0.998	0.989	0.958	0.7056
RMHC-G	0.993	0.755	0.093	0.998	0.962	0.303	0.6846
ERS3	0.990	0.517	0.068	0.998	0.957	0.114	0.6077
GA-E	0.689	0.072	0.031	0.985	0.053	0.067	0.3165
GA	0.626	0.092	0.028	0.359	0.127	0.033	0.2113
RMHC-L	0.031	0.019	0.015	0.054	0.016	0.111	0.0415

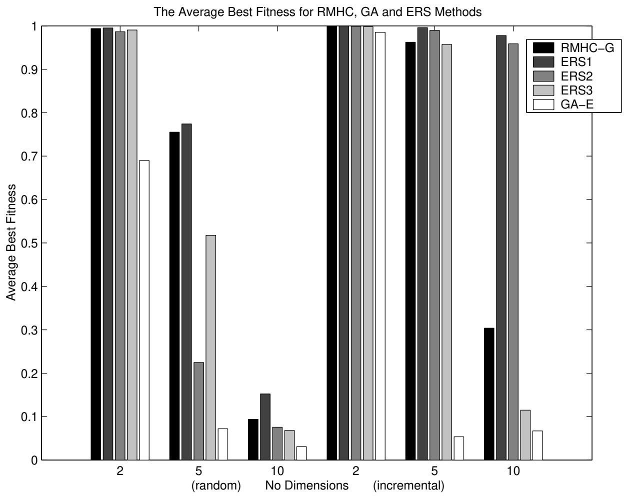

Figure 5: A comparison of the ERS methods to the GA-E and RMHC-G methods

To assess overall performance the average of each row in Table 1 was taken. The most successful was ERS1, with an average fitness of 0.8157, though its result can be dismissed due to the excess fitness evaluations mentioned above. This was followed by ERS2 with a fitness score of 0.7056, closely followed by RMHC with a score of 0.6846 (the difference is not statistically important). ERS3 performed respectably with 0.6077, and the best of the remaining methods, GA-E, had an average fitness of 0.3165.

Overall RMHC-G and ERS2 could hardly be separated in terms of performance, and this suggests that there is scope for further research into combinations of evolutionary and non-evolutionary methods. The ERS methods also showed significant improvement over the GA methods, due to their non-destructive mutation operators promoting continual population diversity (with *no need* to trigger some form of hypermutation). Comparing the ERS3 and RMHC-G results is interesting since it suggests that crossing over two poor individuals in ERS3 regularly yielded an individual that was almost as fit as a globally mutated individual in RMHC. Further work with different types of crossover is needed to establish whether this conclusion holds in general, or at all. If so this may indicate that ERS3's combination of non-destructive mutation, and weaker GA crossover, partially mitigates the hitchhiking phenomenon Mitchell et al. (1993).

6 Conclusions

As far as the authors are aware, the work presented here is the first attempt to unify evolutionary and non-evolutionary methods in dynamic optimisation. The first aim of this work was to answer: "Can RMHC and a GA be combined to produce a method that ensures a continually diverse population?" In this instance it can, by means of continual high levels of non-destructive mutation. The question now is how generalisable these results are.

The second aim was to discover how ERS performed relative to RMHC, and why. The results were good, but mainly because, like RMHC, ERS contains such high mutation. Perhaps this was to be expected since Grefenstette and Cobb noted that random immigrants were useful, but controlling the rate of their mutation had little value (Grefenstette, 1999); and since, in the results above, low levels of mutation were not particularly effective. The application of crossover, even to medium fitness chromosomes, appears to be quite effective, since it alters the values of several dimensions at once – something the mutation operator can not do.

On the basis of the results presented here, a suggested answer to Mitchell *et al*'s question, "When will a GA outperform hill climbing?" is "when it combines crossover with strong non-destructive mutation operators." In any case, EA practitioners — in contrast to those who model genetic and evolutionary processes — should neither dismiss EAs as inferior to RMHC, nor look for failings in the RMHC methodology above that will excuse the GA's performance. More fundamentally perhaps, one might question the need for a separation between EAs and hill climbing at all. Some forms of ES are little more than self-adapting mutation; is this an EA or a form of hill climbing? Is such a dichotomy useful?

7 Further Work

The work presented here is being expanded by:

- Application and comparison of the methods to several, qualitatively different fitness functions, *c.f.* Ursem (2000).
- Comparison to other existing methods, such as ES and the approaches outlined in Section 2.2, particularly the use of age (Ghosh et al., 1998).
- Establishing whether the conclusions hold in general, and if so whether crossing over poor solutions can be used to mitigate the hitchhiking problem.
- Reducing the number of generations before the fitness function changes, until a change occurs every generation and testing the use of memory in the ERS (as briefly described in the text above).

References

Angeline, P. J. (1995). Adaptive and self-adaptive evolutionary computations. In *Computational Intelligence: A Dynamic Systems Perspective*, pages 152–163. IEEE Press, Piscataway, NJ.

Branke, J. (1999a). Evolutionary approaches to dynamic optimization problems – a survey. In *GECCO Workshop on Evolutionary Algorithms for Dynamic Optimization Problems*, pages 134–137.

Branke, J. (1999b). Memory enhanced evolutionary algorithms for changing optimization problems. In *Congress on Evolutionary Comutation CEC'99*.

Cobb, H. (1990). An investigation into the use of hypermutation as an adaptive operator in genetic algorithms having continuous, time dependent nonstationary environments. NRL Memornadum Report 6760 (NCARAI report AIC-90-001).

Cobb, H. and Grefenstette, J. (1993). Genetic algorithm for tracking changing environments. In *Proceedings of the 5th International conference of Genetic Algorithms*, pages 530–532.

Dasgupta, D. and McGregor, D. (1992). Nonstationary function optimisation using the structured genetic algorithm. In *Proceedings of Parallel Problem Solving from Nature (PPSN-2)*.

De Jong, K. (1999). Evolving in a changing world. In *Foundations of Intelligent Systems*, pages 512–519.

Eggermont, J., Lenaerts, T., Poyhonen, S., and Termier, A. (2001). Raising the dead; extending evolutionary algorithms with a case-based memory. In *Proceedings of Second European Conference on Genetic Programming (EuroGP'01)*.

Ghosh, A., Tstutsui, S., and Tanaka, H. (1998). Function optimization in nonstationary environment using steady state genetic algorithms with aging of individuals. In *IEEE International Conference on Evolutionary Computation*, pages 666–671.

Goldberg, D. E. and Richardson, J. (1987). Genetic algorithms with sharing for multimodal function optimization. In *Proceedings 2nd International Conference on Evolutionary Programming*, pages 296–307.

Grefenstette, J. (1992). Genetic algorithms for changing environments. In *Proceedings of Parallel Problem Solving from Nature 2*, pages 137–144.

Grefenstette, J. (1999). Evolvability in dynamic fitness landscapes: a genetic algorithm approach. In *Proceedings of the Congress on Evolutionary Computation (CEC99)*, pages 2031–2038.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems.* The University of Michigan Press, Ann Arbor,MI.

Koza, J. R. (1992). *Genetic Programming.* MIT Press, Cambridge, MA.

Lang, K. (1995). Hill climbing beats genetic search on a Boolean circuit synthesis problem of Koza's. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 340–343, San Francisco, CA. Morgan Kaufmann.

Lang, K., Koza, J. R., and Tahoe, J. K. (1995). Webpages provided by J. K. Tahoe. http://www.genetic-programming.com/jktahoemain. html.

Lewis, J., Hart, E., and Ritchie, G. (1998). A comparison of dominance mechanisms and simple mutation on nonstationary problems. In *Parallel Problem Solving from Nature – PPSN V*, pages 139–148, Berlin. Springer.

Louis, S. J. and Johnson, J. (1997). Solving similar problems using genetic apgorithms and case-based memory. In *7th International Conference on Genetic Algorithms*, pages 481–488.

Louis, S. J. and Xu, Z. (1996). Genetic algorithms for open shop scheduling and re-scheduling. In *International Conference on Computers and their Applications*, pages 99–102.

Mitchell, M., Holland, J. H., and Forrest, S. (1993). When will a genetic algorithm outperform hill climbing. In *Advances in Neural Information Processing Systems*, volume 6, pages 51–58. Morgan Kaufmann Publishers, Inc.

Oppacher, F. and Wineberg, M. (2000). Reconstructing the shifting balance theory in a GA: taking Sewall Wright seriously. In *The Proceedings of the 2000 Congress on Evolutionary Computation (CEC2000)*, pages 219–226.

Ramsey, C. L. and Grefenstette, J. J. (1993). Case-based initialisation of genetic algorithms. In *5th International Conference on Genetic Algorithms*, pages 84–91.

Ursem, R. K. (2000). Multinational GAs: Multimodal optimization techniques in dynamic environments. In *Proceedings of Genetic and Evolutionary Computation Conference (GECCO-2000)*, pages 19–26.

Walker, J. H. and Wilson, M. (2002). How useful is lifelong evolution in the physical world. Technical report, University of Wales, Aberystwyth. UWA-DCS-02-040.