Comparison of Steady State and Generational Genetic Algorithms for Use in Nonstationary Environments

Frank Vavak
Faculty of Computer Studies & Mathematic
University of the West of England
Bristol BS16 1QY, UK
f_vavak@btc.uwe.ac.uk

Terence C. Fogarty
Faculty of Computer Studies & Mathematic
University of the West of England
Bristol BS16 1QY, UK
tcf@btc.uwe.ac.uk

Abstract

The objective of this study is a comparison of two models of the genetic algorithm, the generational and incremental/steady state genetic algorithms, for use in nonstationary/dynamic environments. It is experimentally shown that choice of a suitable version of the genetic algorithm can improve its performance in such environments. This can extend ability of the genetic algorithm to track environmental changes which are relatively small and occur with low frequency without need to implement an additional technique for tracking changing optima.

1 Introduction

The genetic algorithm is a proven search/optimisation technique [Holland 1975] based on an adaptive mechanism of biological systems. In our previous work we showed that the genetic algorithm is a suitable on-line optimization method to balance the load of the presses in a sugar beet pressing station [Fogarty,Vavak,Cheng 1995] and in load balancing of multiple burner boiler [Vavak,Fogarty,Jukes 1995]. Because both mentioned applications are the time varying systems, a possibility of tracking changing optima has to be considered. If the system is subjected to abrupt changes the genetic algorithm has to be periodically restarted (i.e. a new search is initiated from a random starting point in the search space) or a suitable method for tracking changing optima has to be implemented (e.g. [Cobb,Grefenstette 1993]). Nevertheless, the performance of a genetic-algorithm based control system can be improved by a selection of a suitable version of the genetic algorithm. If the environmental changes are relatively small and occur with low frequency (e.g. system parameters drifting) the selection of a suitable model of the genetic algorithm can extend its ability to track such environmental changes without need to implement additional tracking techniques.

Work presented in this paper is a comparison of two models of the genetic algorithm, the generational (GGA) and the incremental (IGA) one, for use in nonstationary environments. Because the chromosomes in the applications mentioned above are evaluated directly (through experimentation), an important consideration when comparing the models of the genetic algorithm was not only the speed of convergence to the global optimum and the off-line performance but the on-line performance as well.

2 The Specific Problem

The problem used in this study to examine the generational and the incremental genetic algorithms for nonstationary environments does not represent any particular application and was chosen so that analysis of the results was easy and clear.

The population of the genetic algorithm consists of binary strings. The evaluation function returns a value equal to the number of the corresponding alleles/bits which are identical in a given chromosome and a predefined "template chromosome" (i.e. bit matching task). Change of a selected number of bits of the template simulates the environmental change - the Hamming distance between the "old" and "new" locations of the optima is given by the number of the flipped bits. Length of the chromosomes used in our tests is 40 bits, giving a total search space of 2^{40} points.

0-7803-2902-3/96/$4.00©1996 IEEE

3 The Two Models of Genetic Algorithm

The generational genetic algorithm (the "Standard Genetic algorithm" [Cobb,Grefenstette 1993]) creates new offspring from the members of an old population using the genetic operators and places these individuals in a new population which becomes the old population when the whole new population is created [Goldberg 89, De Jong 1992]. The incremental/steady state genetic algorithm [Whitley, Kauth 1988] is different to the generational model in that there is typically one single new member inserted into the new population at any one time. A replacement/deletion strategy defines which member of the population will be replaced by the new offspring. In this paper we examine two standard replacement strategies - deleting the oldest and deleting the worst member of the population.

Two different methods for picking two parents to create an offspring were tested. In the first set of the tests a roulette wheel sampling mechanism and a proportional selection was used for both the incremental and generational genetic algorithm (the fitness values were scaled: fitness = current fitness - fitness of the worst chromosome). The second set of experiments implemented Baker's "standard universal sampling" (SUS) [Baker 1987] and linear ranking selection method for the generational genetic algorithm. The fitness of a chromosome f(i)=s-2(i-1)(s-1)/(N-1), s=2 and i={1.N} where N is the population size. The standard universal sampling does not suffer from a sampling error (i.e. is not source of extra noise for the generational genetic algorithm) unlike the roulette wheel sampling [Baker 1987]. The incremental genetic algorithm uses a tournament selection with the tournament size 2. The better chromosome wins with a probability 1. This results in comparable selection pressures for both models of the genetic algorithm [Hancock 1994].

Both the generational and incremental genetic algorithm used for our tests implement one point crossover.

4 Results

All the experiments were run for the population size 100 and the results were averaged over 50 runs with different seed values. The same set of the random seed values was used for runs of the genetic algorithms with different combinations of mutation rates (0.01; 0.005; 0.002767; 0.001; 0.0005), crossover rates (0.8; 1.0), magnitudes of environmental change and sampling/selection methods.

Comparison of the on-line and off-line performance for both models of the genetic algorithm shows that the difference in the performance is consistent and statistically significant across the spectrum of the parameter setting combinations.

4.1 Starting with a Random Population

The following tables show the typical test results after re-start of the genetic algorithm for the probability of crossover 1.0 and mutation rate 0.002776 - i.e. 1.75 / (sqrt(chromo.length) * pop.size) [Back 1991]. This situation is relevant to the environmental changes which can occur while the population diversity is still high.

	no. of eval. to success	worst after 20000 eval.	no.of best after 20000 evaluations
IGA(del.worst)	584	39.3	99.9
IGA(del.oldest)	1054	38.4	84.8
GGA	1725	38.3	83.6

IGA & GGA - roulette wheel & proportional selection

	no. of eval. to success	worst after 20000 eval.	no.of best after 20000 evaluations
IGA(del.worst)	653	39.99	99.9
IGA(del.oldest)	863	39.9	88.4
GGA	1225	39.9	89.1

ICG - tournament selection
GGA - standard universal sampling, linear ranking

It is clear that the incremental model outperforms the generational genetic algorithm in terms of the convergence speed (i.e. the number of evaluations before the optimum is found is smaller). The incremental algorithm using deleting the worst replacement strategy provides also better results than the other two versions as far as number of the best individuals in the population is concerned. It gives nearly 100% converged population due to the higher selection pressure resulting from the replacement strategy used.

4.2 Effect of an Environmental Change on a Converged Population

In this section we describe the experiments when the environmental change occurred while the population was fully converged. It is a kind of environmental change relevant to the environment changes common in the applications mentioned in section 1 because they are more likely to happen when the population of the genetic algorithm is converged.
The genetic algorithm can then exhibit tracking ability primarily thanks to the presence of the mutation mechanism. The tests were run for the previously described combinations

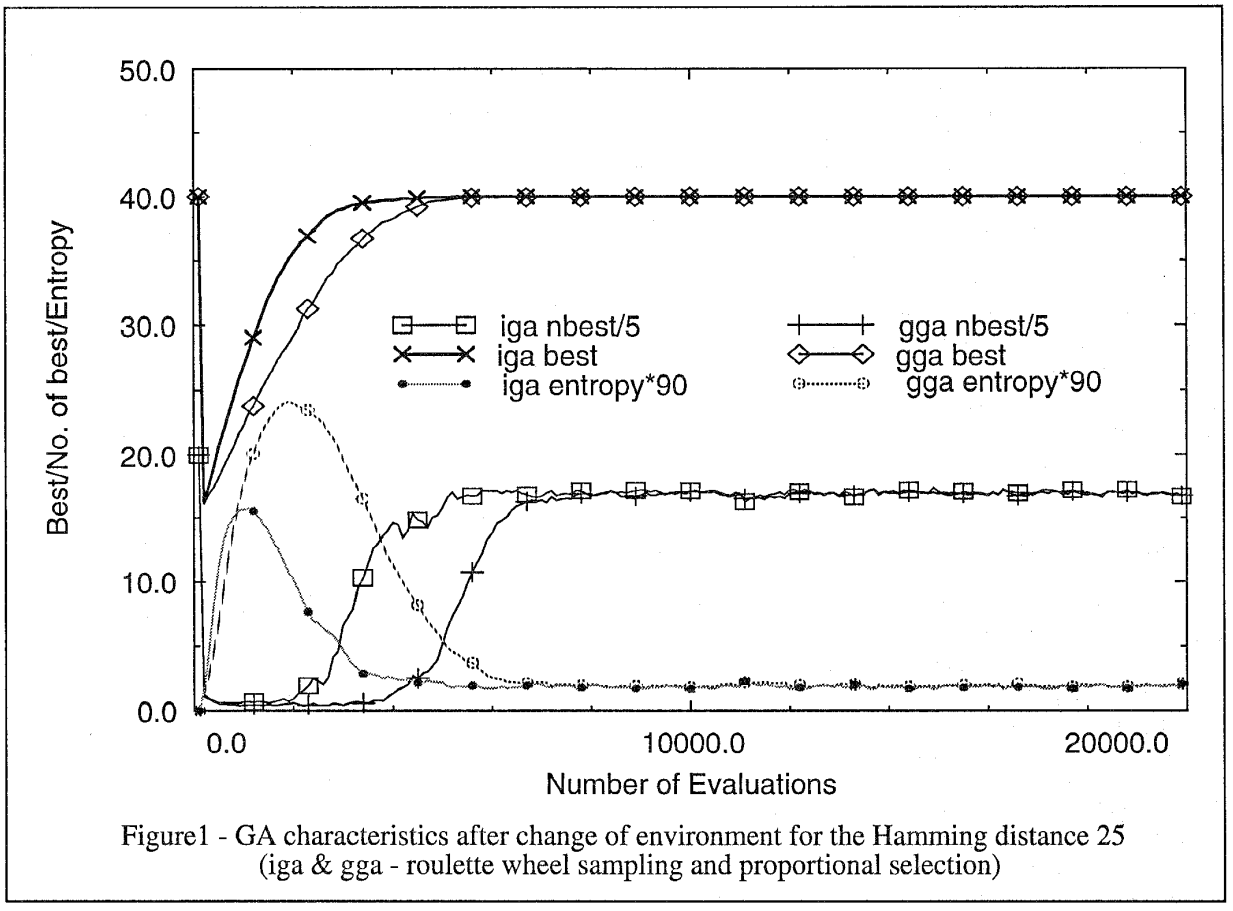

Figure1 - GA characteristics after change of environment for the Hamming distance 25 (iga & gga - roulette wheel sampling and proportional selection)

of mutation and crossover rates. The Hamming distance of the environmental change was gradually set to 5, 10, 15, 20, 25, 30, 35 and 40.

Test data obtained is consistent for all combinations of the parameter settings and sampling/selection methods. Figure 1 illustrates the typical results obtained. Corresponding to the results from the previous section, the incremental genetic algorithm (deleting the oldest member of the population strategy) outperforms the generational genetic algorithm. The figure 1 does not show the characteristics for the incremental genetic algorithm using deleting the worst member of the population replacement strategy because this model of the genetic algorithm cannot track any changes of the environment. It is obvious from the mechanism of the replacement, that the only chromosome repeatedly re-evaluated after the change of the environment is the worst one. The fitness values of no other member of the converged population is affected by the environmental change. The fitness values thus stay outdated indefinitely.

Figure 1 also indicates the values of population entropy [Davidor,Ben-Kiki 1992] which is a measure of disorder in the population (i.e. entropy = 0 for a fully converged population). It can be seen that the generational genetic algorithm introduces higher diversity into the population than the incremental genetic algorithm before it finds the optimal solution. This can cause increased disturbances to the system controlled if the genetic algorithm is, for example, used for on-line optimisation.

Figure 2 shows relation between the on-line/off-line performance and the Hamming distance of the environmental change for the mutation rate 0.002767.

5 Conclusions

Our experiments with a variety of parameter settings for both models of the genetic algorithm showed that the incremental genetic algorithm with the "deleting the oldest" replacement strategy is superior to the generational genetic algorithm as far as the on-line and off-line performance is concerned. It can extend ability of the genetic algorithm to track environmental changes, which are relatively small and occur with low

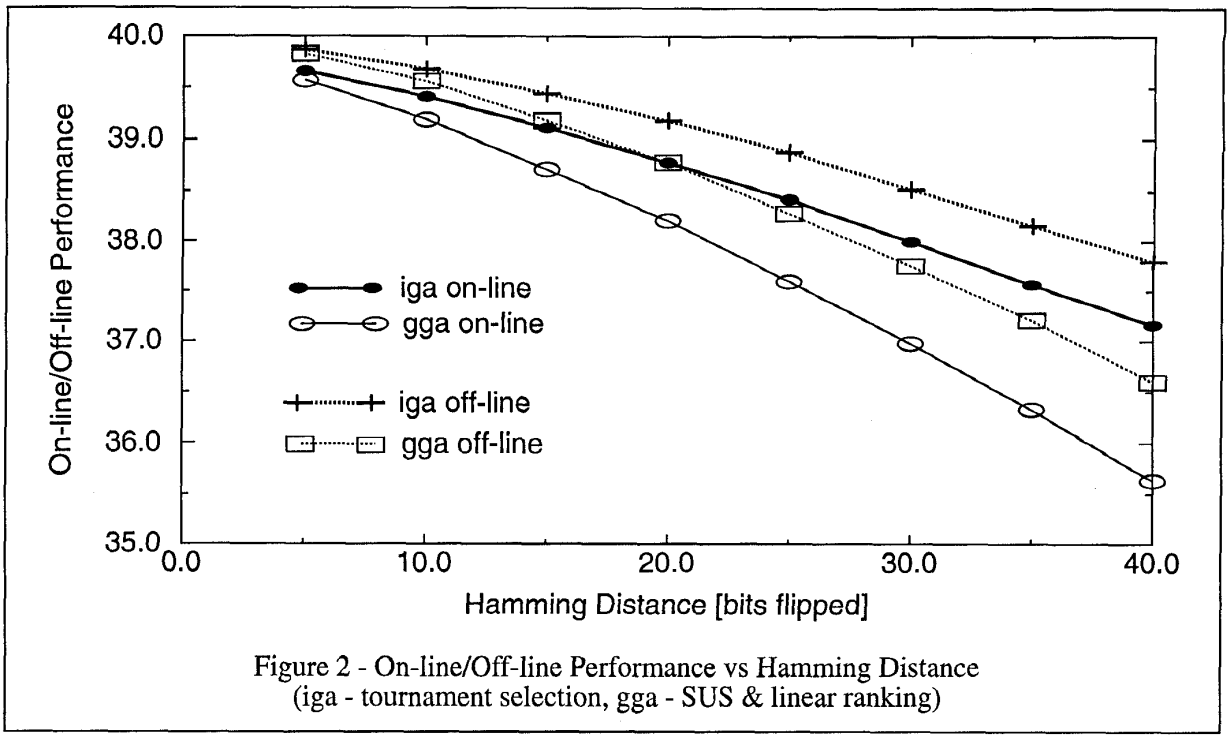

Figure 2 - On-line/Off-line Performance vs Hamming Distance
(iga - tournament selection, gga - SUS & linear ranking)

frequency without need to implement an additional technique for tracking changing optima. Provided the incremental genetic algorithm is used for on-line optimization, smaller diversity introduced into the population after an environmental change decreases the disturbances acting on the system controlled.

The better performance of the incremental genetic algorithm can be explained by the fact that in the incremental genetic algorithm an offspring is immediately used as a part of the mating pool, making a shift towards the optimal solution possible in a relatively early phase of the optimization process.

References

Back T (1991) "Self Adaptation in Genetic Algorithms" in "Towards a Practice on Autonomous Systems" ed. Varela and Bourgine (MIT Press 1992).

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

Fogarty T C, Vavak F, Cheng P (1995) "Use of the Genetic Algorithm for Load Balancing in the Process Industry" - 6th International Conference on GA (Morgan Kaufmann Publishers,Inc.).

Vavak F, Fogarty T C, K Jukes (1995) "Application of the Genetic Algorithm for Load Balancing of Sugar Beet Presses" -1st International Mendel Conference on GA, (PC-DIR Publishing, Brno).

Cobb H, Grefenstette J(1993) "GA for Tracking Changing Environments" - 5th International Conference on GA (Morgan Kaufmann Publishers,Inc.).

Goldberg D E (1989) "Genetic Algorithms in Search, Optimisation and Machine Learning" (Addison Wesley).

De Jong K A (1992) "Are Genetics Algorithms Function Optimizers?" - Parallel Problem Solving From Nature 2 (Elsevier Science Publisher).

Whitley D, Kauth J (1988) ".GENITOR: A different Genetic Algorithm" in Proc. of the Rocky Mountain Conf. on Artificial Intelligence" , Denver.

Davidor Y, Ben-Kiki O (1992) "The Interplay Among the Genetic Algorithm Operators: Information Theory Tools Used in a Holistic Way" - Parallel Problem Solving From Nature 2 (Elsevier Science Publisher).