Inactive subpopulations improve adaptability and diversity

Dominik Herrmann
dominik at herrmann-net.de

School of Computer Science & Informatics
University College Dublin

Abstract. Real-world problems involve dealing with dynamic environments. Often, naïve implementations of genetic algorithms fail here, because the optimum solution is a moving target. This paper proposes an extension to the concept of the population in order to increase its adaptability. Therefore, the population is segmented into subpopulations, whereas only one of them is actively engaged at a given time. Various experiments demonstrate how to prevent undesired convergence through exchanging individuals between subpopulations. It is shown that this macromutation can improve flexibility, delay convergence and confer an adaptive advantage in a dynamic environment if the right set of parameters is used.

1 Introduction

Genetic algorithms are supposed to find the optimum solution to a problem through evolutionary pressure on a population of individuals representing the specific problem. The algorithm resembles evolution in nature by employing crossover and mutation operators. This powerful concept was proposed by Holland [3]. Since then, it has undergone much research in order to understand the inner workings of optimisation of static and dynamic problems.

Optimisation problems in dynamic environments are a particularly difficult task - even for genetic algorithms. In contrast to static problems the fitness function to optimise is changing throughout time. Therefore, it is not sufficient to find the optimum solution only once. Instead, every change has to be recognised, so that the solution can be adjusted to the new environment. A number of papers discuss how to address this problem when a change has been detected, popular approaches are Hypermutation [1] and Variable Local Search [7].

This paper will not discuss *how to detect change* but concentrate on the ability of a genetic algorithm to track the optimum solution of a dynamic fitness function. Previous research mainly dealt with this problem by embedding inactivity or redundancy into the algorithm [2]. Levenick has shown that introns, i.e. inactive parts of the genome, are able to confer an adaptive advantage over regular individuals in dynamic environments [6]. Another approach is to integrate redundancy on the level of the population. This idea led to parallel genetic algorithms [5] and island models [4], which have a similar approach like this paper.

Because the existing approaches are often very sophisticated, it can be difficult to understand the underlying workings of the algorithm. Therefore, it is worthwhile to study the behaviour of genetic algorithms in dynamic environments on a very basic level. This paper aims to evaluate whether it is possible to improve the performance of a genetic algorithm by splitting its population into an active and an inactive segment. Therefore, a number of experiments was performed in order to compare this approach to a traditional algorithm. A classic genetic algorithm will be compared to a modified one, which is able to exchange individuals between two subpopulations, in context of a simple dynamic fitness function. It will be evaluated whether immigrants confer adaptive advantage and are able to increase diversity.

A brief description of dynamic environments and inactive subpopulations is provided in section 2. Section 3 describes the setup and the overall methodology for the experiments, whereas sections 4 and 5 elaborate on the results of extensive experiments on adaptive advantage and a short study of diversity. Finally, section 6 presents some ideas for further research and summarise the findings in section 7.

2 Inactive subpopulations in dynamic environments

This section provides a brief description of the application of genetic algorithms in dynamic environments. Furthermore, it introduces the notion of *inactive subpopulations*. Finally, it states two hypotheses, which will be verified with the help of three experiments (cf. sections 4 and 5).

2.1 Characteristics of dynamic environments

Dynamic environments can be classified by numerous parameters. For the purpose of this paper the following are of relevance:

- continuous changes versus abrupt changes (in terms of time)
- small changes versus large-scale changes (in terms of the optimisation problem)
- recurrence or non-recurrence of change (repeated pattern or unique sequence)
- change rate, i.e. the absolute time interval between changes (number of generations)

Note that this paper only focuses on recurring changes with a constant rate of change.

The structure of genetic algorithms as presented in section 1 allows them to adapt to continuous small changes. Large-scale changes of the fitness function during a run can pose a problem for them, though.

2.2 Success factors of genetic algorithms

Generic problems Holland identifies two driving characteristics of genetic algorithms*, exploration* and *exploitation*. Exploration refers to the ability of an algorithm to exhaustively search for the optimum solution in the entire fitness landscape in order to find the global optimum and not only a local one. Exploitation addresses its effectiveness in finding the actual optimum value once a nearby, almost optimum solution has been found.

These two conflicting forces lead to the problem of *premature convergence* [9]. In this context convergence indicates overall homogeneity across the population, i.e. that a large part of the population is concentrated on one particular solution (trying to exploit it most effectively). At that stage of a run it is usually impossible to divert the search to other parts of the fitness landscape (leading to poor exploration).

Dynamic environments Previous research suggests two important concepts for the performance of a genetic algorithm in dynamic environments: *adaptability* [4] and *diversity* [6]. An algorithm has to be able to detect and track changes of the fitness landscape and adapt its interior structure to the new surroundings. This can be facilitated by maintaining some amount of diversity or redundancy.

When it comes to dynamic fitness functions premature convergence is particularly problematical. Once a large amount of individuals has converged on the seemingly optimum solution it can have great difficulty in the event of change. Therefore, preventing premature convergence is an especially important target in this context.

2.3 Effectiveness of inactive subpopulations

For the purpose of this paper, a number of experiments was conducted, utilising a genetic algorithm with two distinct populations. I will call the proper population the *active* one and the second population the *inactive* one. This term refers to the fact that only the individuals in the active population are exposed to selection pressure based on their current fitness value. The inactive subpopulation is allowed to breed independently by always selecting *random* parents for the next generation.

In both populations the basic genetic operators mutation and crossover are applied during breeding. The approach presented in this paper additionally employs a *macromutation*: After a certain number of generations some amount of individuals is exchanged between subpopulations. Thus, new genetic material is injected into the active subpopulation, which may be beneficial for solving the optimisation problem. Note that the population size is not affected by the exchange as the migrated individuals replace existing individuals in the population.

It has been shown [5] that inactive subpopulation can work as a memory or cache. Apart from that, it might be an effective means to increase diversity in the active subpopulation and thus delay premature convergence.

The exchange process is controlled by several parameters: The *migration rate* determines how frequently individuals migrate. The extent of the influence of an exchange operation is addressed by the *migration amount* (in % of the population size). Another means for tuning the exchange process is the *selection method* applied to determine which individuals will be migrated (the best individuals or randomly selected ones) and which get replaced by the migrants (the worst individuals or random ones). Finally, the structure of the exchange process itself can be altered, e.g. migrating individuals only from the inactive to the active population but not vice versa. The influence of the aforementioned parameters will be evaluated by simple experiments.

2.4 Hypotheses

Hypothesis 1: In a dynamic environment, migrating individuals between active and inactive subpopulations can confer adaptive advantage (cf. section 4).

In previous studies premature convergence was identified as one of the main obstacles when it comes to flexibility and adaptability [6]. Therefore, it is important to test another hypothesis:

Hypothesis 2: In a dynamic environment, migrating individuals between active and inactive subpopulations will delay convergence and retain diversity (cf. section 5).

3 Methodology

3.1 Parameters and setup

All experiments were run 30 times in order to get meaningful average results. The genetic algorithm from the software library ECJ v15[1], which was slightly extended for the experiments, was used. Table 1 lists the basic parameters for the evolutionary process. Appropriate values were determined in a series of preliminary experiments.

Table 1. Common genetic algorithm parameters for all experiments

Parameter	Value
xover method and rate	ECJ anypoint / 0.7
mutation rate	0.01
selection method	tournament selection, size 2
random number generator	ECJ MersenneTwister
seed of first run	4358
number of generations	1000
genome size	100 bits

3.2 Fitness function and metrics

The fitness function used in the experiments is an extension of the popular *One-Max* fitness function. *One-Max* calculates the fitness of an individual by just counting the bits set to 1. Thus, the *ideal individual* has all of its bits set to 1. In the context of optimisation problems this can be interpreted as maximising the amount of 1s in the bit string.

For the experiments a simple dynamic environment will be simulated by the so called *Zero-One-Max* fitness function. It switches from *One-Max* (maximising 1s) to *Zero-Max* (maximising 0s) and vice versa on a regular basis. The abruptness and the recurrence rate of the change can be adjusted to evaluate the behaviour in different scenarios. For this paper two distinct scenarios were created, namely *continuous change* and *abrupt change*.

In experiment 1 (cf. section 4.1) the dynamic fitness function exerts *abrupt change* by alternating between the two extreme cases. The change interval is 100. As the experiments run for 1000 generations there will be 9 changes in total. Between these changes the fitness function stays constant.

In experiment 2 (cf. section 4.2) there are more (49) smaller changes with shorter intervals between the individual changes (changes occur every 20 generations) simulating *continuous change*. After a change 20% of the genome of an ideal individual has to be modified in order to become ideal again. The behaviour of the fitness functions is outlined in figure 1. The graph shows the required relative amount of 1s for an ideal individual over time.

In experiments 1 and 2 the adaptability of the algorithms is evaluated using the following metrics:

IdealIndividualsFound refers to the effectiveness of an algorithm. It indicates how often at least one ideal individual was found after one change but before the next change. The values are averaged over all the runs. The optimum value of this metric for experiment 1 is 10, for experiment 2 it is 50.

[1] http://cs.gmu.edu/~eclab/projects/ecj/

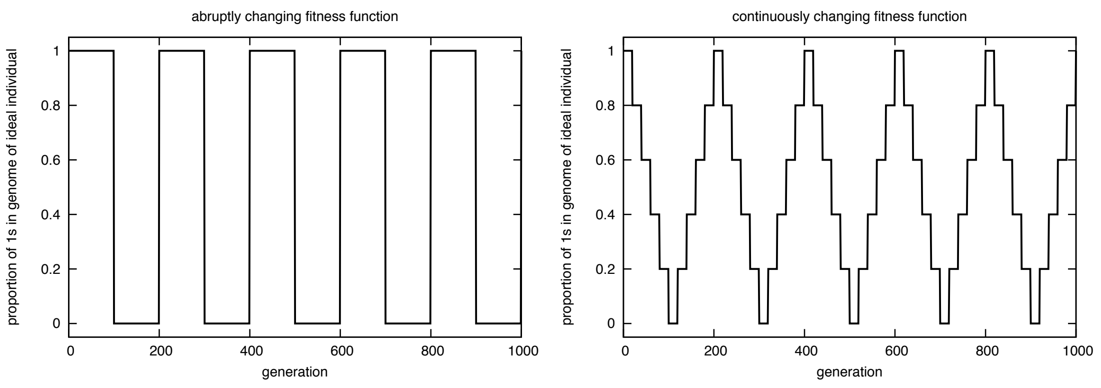

Fig. 1. Dynamic fitness functions employed in experiment 1 and 2

FirstIdealIndividual is a measure of efficiency. It shows how fast an ideal individual was found after a change of the fitness function. It is calculated as the absolute generation number of the first occurrence of an individual with relative fitness 1.0 modulo the change rate. Therefore, smaller values are better whereas the worst value is the value of the change rate (when the ideal individual is found just in the last generation before the next change is due).

Experiment 3 aims to measure the diversity in the populations by using the following metrics:

StandardDeviationOfFitness measures diversity expressed in the fitness of all individuals of a population. Converged populations usually consist of individuals with the same fitness whereas diverse populations offer a wide range of fitness values. Higher values of this metric indicate a more diverse population. Care has to be taken interpreting this metric because individuals with a very unequal genome can have the same fitness. Therefore, an additional genome-based metric is necessary.

GenomeVariability intends to measure the diversity in the genome of a population. This is achieved by iterating over all individuals and separately calculating the statistical variance for each bit position. The metric is the sum of those bitwise variances. Higher values indicate a more diverse population.

Where appropriate the results were supported by a statistical measure:

p-value is a statistical indicator of significance which is returned by the Student's t-test. The p-value gives the probability that the fact that two distributions have identical arithmetic means $\overline{x}$ is due to pure coincidence. Usually, a level of significance of $\alpha = 0.05$ is used as threshold. Therefore, if the p-value is smaller than α the two samples are said to be significantly different in terms of their arithmetic means

4 Evaluation of adaptive advantage

This section presents two experiments which compare the adaptability of a classic genetic algorithm (with one uniform population) with a genetic algorithm utilising two populations (one active, one inactive). The two experiments evaluate two migration strategies each, namely *continuous migration* and *abrupt migration*.

In a preliminary experiment the classic algorithm was invoked on the dynamic fitness function with different population sizes. This constitutes the base case. After that, the modified algorithm was invoked on the problem. During the run the IdealIndividualsFound and FirstIdealIndividual were recorded. Exchanging individuals between populations would confer adaptive advantage if the modified algorithm found the FirstIdealIndividual faster than the classic algorithm.

4.1 Experiment 1: abrupt changes

For this experiment the abrupt *Zero-One-Max* function with a change rate of 100 was applied (cf. section 3.2). All runs found the maximum of 10 ideal individuals per run. The results for the base case are shown in table 2. It can be observed that the performance of the classic algorithm increases with population size, which is not surprising.

Table 2. Base case: performance of classic genetic algorithm increases with population size

Population size	**100**	**200**	**400**	**600**	**800**	**1000**	**2000**
FirstIdealIndividual (average)	66.78	58.79	54.33	53.06	52.06	51.35	49.71

Continuous migration For comparison, the modified algorithm was run with a migration rate of 2 and a migration amount of 3% of the population. Furthermore, the *individuals to be sent from the inactive population* were chosen with tournament selection (selecting the most appropriate individuals under the current circumstances, see section 6 for more discussion of this scheme) while the *individuals sent from the active population* were determined randomly. Again, all ideal individuals were found, and the metric FirstIndividualFound follows a similar trend: All values are consistently better (lower) for the same population size. Though, one should take into account that the modified algorithm has two populations with a doubled total size. Therefore, the comparison has to be performed against a regular algorithm with a population of twice the size. Table 3 shows the results of this benchmark.

Table 3. Benchmarking the modified algorithm for different population sizes

Total (active+inactive) population size	**200**	**400**	**800**	**2000**
FirstIdealIndividual regular algorithm	58.79	54.33	52.06	49.71
FirstIdealIndividual modified algorithm	63.42	55.65	51.35	48.52
p-value	0	1.58E-5	5.81E-3	8.90E-8
Algorithm with better performance	regular	regular	modified	modified

The results show that the act of exchanging individuals between the two subpopulations can lead to a significant adaptive advantage for large population sizes when 3% of the population are exchanged every 2 generations. This result is noteworthy because it suggests that continuous migration is able to cope with abrupt changes.

For a more detailed analysis how the two parameters migration rate and migration amount influence the performance, additional tests were carried out with a fixed population size of 800 (400 active, 400 inactive individuals respectively).

For a thorough analysis the value of the migration amount was varied between 0.5% and 20%, the value of the migration rate between 1 and 8. The results of these runs are graphed in figure 2. Note that the graph shows the relative performance in terms of FirstIdealIndividual, which means that a value of +1% means that on average the modified algorithm found the first ideal individual 1% faster than the classic algorithm.

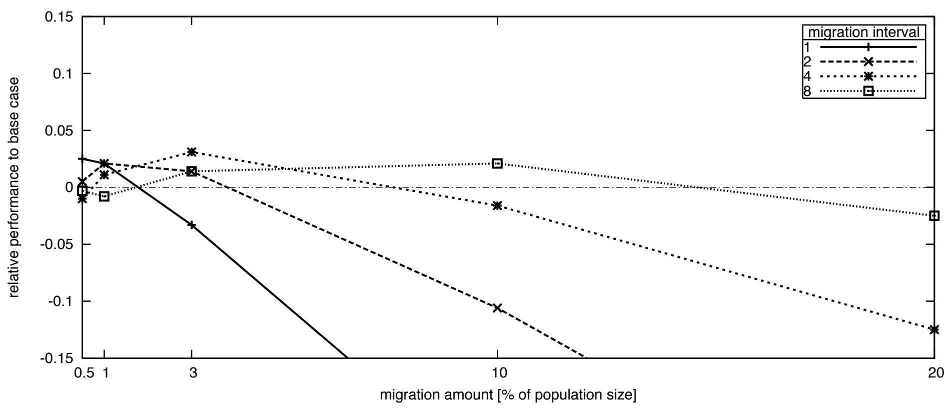

Fig. 2. Relative performance (FirstIdealIndividual) of modified algorithm in relation to classic algorithm

As figure 2 indicates there is a number of scenarios (migration interval, migration amount), namely (1, 0.5%), (1, 1%), (2, 0.5%), (2, 1%), (2, 3%), (4, 1%), (4, 3%), (8, 3%), (8, 10%), which allow the modified algorithm to outperform the classic one. Further analysis has shown, that the difference of the FirstIndividualFound metric is significant on a level of at least $\alpha = 0.05$. Furthermore, it can be observed that it is not worthwhile to exchange a high amount of individuals in short intervals. With increasing intervals a larger part of the population has to be exchanged in order to achieve a positive impact on the performance.

Nevertheless, the above result cannot verify whether the improvements are really due to the inactive population or just by exchanging individuals of some sort. Therefore, a comparison to completely random individuals is necessary. Figure 3 presents the results of these runs graphically. The graph shows the same trend, only this time the performance gain is clearly visible. Therefore, according to the results inserting completely random individuals instead of individuals from the inactive population leads to a significantly higher performance.

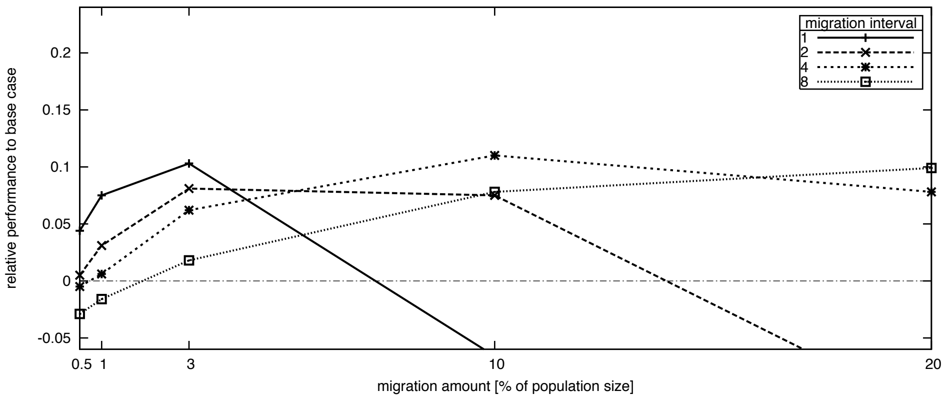

Fig. 3. Relative performance (FirstIdealIndividual) of randomly injected individuals in relation to classic algorithm

Summing up, this experiment has shown that the individuals from the inactive population confer a small adaptive advantage when the continuous migration strategy is used. Though, the source of the advantage is not the inactive population but rather the act of injecting foreign individuals.

One could assume that the continuous migration strategy performs better when it is applied to a continuously changing fitness function. This issue will be addressed later in the second experiment (cf. section 4).

Abrupt migration In the first part of this section it has been shown that there is only a small benefit in applying continuous migration to an abruptly changing fitness function. Abrupt migration – taking place in the generation after the event of change – may be a more appropriate approach. To assess this strategy the experiment was modified: The migration amount was varied between 1% and 100% whereas the migration always took place right after a change was detected (i.e. migration rate of 100 generations). Note that this ensures that each change is “detected” immediately (which may not be always possible in reality, though). Again, a population size of 400 active and 400 inactive individuals was used.

The results of this experiment are shown in figure 4 in terms of relative performance in comparison to the base case of a single, uniform population of 800 individuals.

The graph exhibits a clearly visible trend – higher migration amounts offer better performance. All migration amounts over 1% perform significantly better than the classic algorithm. Furthermore, the graph contains the relative performance values for randomly inserted individuals. Although the random individuals still perform better than uniform populations they fall behind in comparison with individuals which migrated between the two populations.

4.2 Experiment 2: continuous changes

This experiment has a similar structure to the first one. Now, the continuously changing *Zero-One-Max* function is applied (cf. section 3.2). Preliminary experiments have shown that very small changes to the fitness function didn’t pose a problem for the classic algorithm – it was able to follow this royal road [8] and always found an ideal individual at once. Therefore, the fitness function used in this experiment changes its optimum value every 20 generations by 20%.

In contrast to the first experiment, the algorithms didn’t find all ideal individuals (50 in total) any more – the interval between two changes was too short to allow for adaptation to the changed fitness

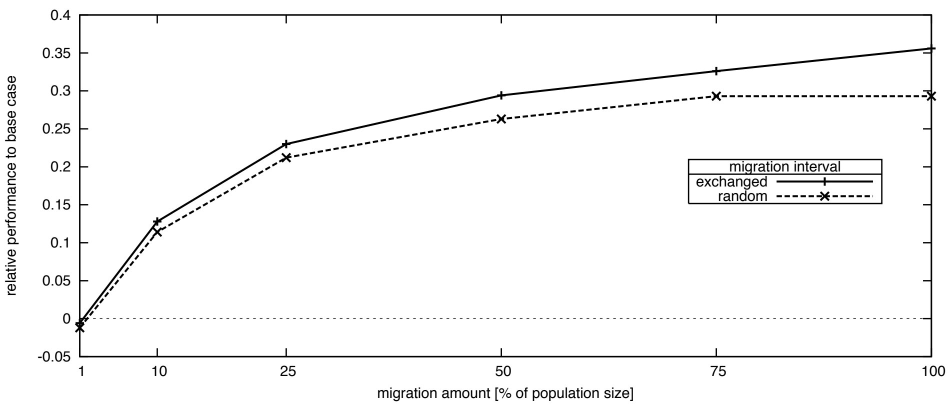

Fig. 4. Relative performance (FirstIdealIndividual) of abrupt migration in comparison to classic algorithm

function. The results of the classic algorithm are shown in table 4. Note that in this experiment two fitness measures are provided, namely IdealIndividualsFound, which indicates the effectiveness of the search, and FirstIdealIndividual, which refers to the efficiency. According to the results the IdealIndividualsFound in a run on average increase with bigger populations. The generation the FirstIdealIndividual is found decreases at first with growing population size, but for populations larger than 600 individuals it increases again.

Table 4. Continuous change: performance of classic algorithm for various population sizes

Population size	**100**	**200**	**400**	**600**	**800**	**1000**	**2000**
IdealIndividualsFound (average)	40.13	41.67	43.60	45.00	46.20	47.07	48.67
FirstIdealIndividual (average)	6.54	6.13	6.12	6.16	6.32	6.38	6.35

Continuous migration The evaluation of adaptability was performed analogous to the first experiment. At the beginning, the modified algorithm was configured to perform continuous migration with migration rate of 2, migration amount of 3% and the same selection methods like described in section 4.1. This time, the comparison between the classic and the modified algorithm are based on two metrics. The results of the experiment are shown in table 5.

Table 5. Benchmarking the modified algorithm for different population sizes

Total (active+inactive) population size	**200**	**400**	**800**	**2000**
IdealIndividualsFound regular algorithm	41.67	43.60	46.20	48.67
IdealIndividualsFound modified algorithm	40.23	41.10	42.97	45.07
p-value	1.49E-5	7.71E-11	2.53E-13	3.85E-15
Algorithm with better performance	regular	regular	regular	regular
FirstIdealIndividual regular algorithm	6.13	6.12	6.32	6.35
FirstIdealIndividual modified algorithm	5.67	5.14	5.14	5.43
p-value	2.49E-5	3.66E-14	2.28E-18	5.05E-18
Algorithm with better performance	modified	modified	modified	modified

The modified algorithm performs consistently worse than the classic one in terms of IdealIndividualsFound. On the other hand, the ideal individuals it did find, emerged significantly faster. Though, there is an inherent dependency between these two metrics, and FirstIdealIndividual has to be interpreted carefully. Whether this behaviour is satisfactory depends on the problem at hand, of course.

Again, a detailed analysis for a fixed population of total size 800 was performed in order to determine the influence of the parameters migration rate and migration. In this case, the modified algorithm performs worse for all combinations of the parameters (cf. figure 5). The graph shows a vague falling trend for higher migration amounts.

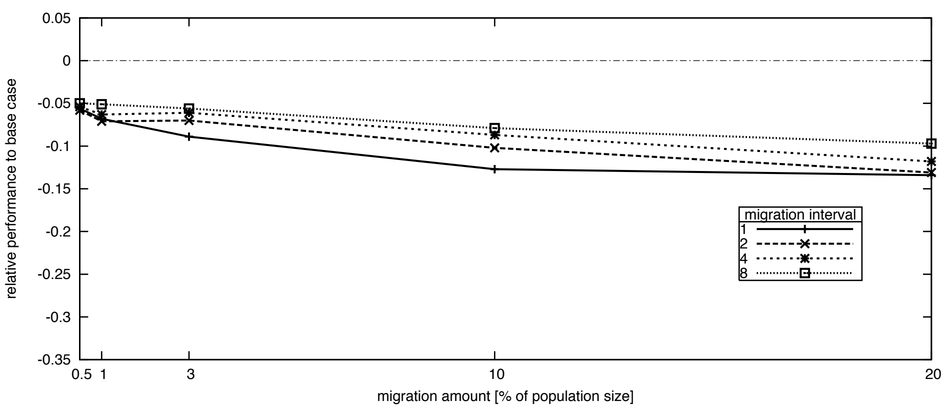

Fig. 5. Relative performance (IdealIndividualsFound) of modified algorithm in relation to classic algorithm

Interestingly, the performance of *randomly selected* immigrants is very similar to individuals chosen from the inactive subpopulation. Table 6 shows that a statistical analysis of the averaged results for various migration amounts offers no significant advantage for exchanged individuals. This finding indicates that solving this specific dynamic fitness function does not profit from external genetic material. Instead, the injected genetic material slows down evolution.

Table 6. Relative performance of random individuals in relation to migrated individuals

Migration amount	**0.5%**	**1%**	**3%**	**10%**	**20%**
IdealIndividualsFound migrated individuals	43.69	43.28	43.00	41.63	40.65
IdealIndividualsFound random individuals	43.51	43.48	42.96	41.61	40.54
p-value	0.34	0.25	0.83	0.90	0.40
Algorithm with better performance	-	-	-	-	-

In this experiment migrating individuals between populations did not confer adaptive advantage. To be sure, the efficiency of finding the new optimum solution was increased, but this was only achieved at the cost of a lower effectiveness.

Abrupt migration Applying abrupt migration with long migration intervals to a continuously changing fitness function seems like the wrong approach. In order to support this suggestion, the tests from the first experiment were rerun (migration amount: between 1% and 100%, migration interval: 100 generations, population size: 400 active and 400 inactive individuals).

Table 7. High migration intervals lead to poor performance of modified algorithm

Migration amount	**1%**	**10%**	**25%**	**50%**	**75%**	**100%**
IdealIndividualsFound (migrated individuals)	43.73	43.37	42.80	41.37	40.20	40.00
IdealIndividualsFound (random individuals)	43.73	43.33	42.90	41.37	40.07	40.00

The results of this experiment are shown in table 7. Obviously, neither migrated individuals nor randomly injected individuals are able to improve the performance, which is due to the wrong migration interval. With increasing migration amounts the performance decreases further because more of the problem knowledge incorporated in the proper population is destroyed on immigration.

4.3 Summary of analysis on adaptive advantage

This section evaluated hypothesis 1, i.e. migrating individuals between active and inactive subpopulations can confer adaptive advantage in a dynamic environment. On the basis of the experiments performed this hypotheses could only be established for applying abrupt migration to an abruptly changing fitness function.

While it is conceivable that continuous migration is not able to cope with an abruptly changing environment, it is rather surprising that it also fails when facing a continuously changing fitness function. However, it has to be noted that the reason for the poor performance is different. In experiment 1 the continuous migration offered some adaptive advantage, but it was outperformed by the injection of random individuals. On the other hand, in experiment 2 neither migrated nor random individuals were able to achieve performance gains employing any of the two migration strategies.

Finally, it was observed that applying the wrong migration strategy to a dynamically changing fitness function is likely to decrease the performance of an algorithm. Therefore, a rough understanding of the dynamics of the fitness function is required for utilising the benefit of migration.

5 Evaluation of diversity

In this section hypothesis 2, that migrating individuals between active and inactive subpopulations can delay convergence and retain diversity, will be evaluated. The setup is the same as in experiment 1, using the abruptly changing fitness function. In table 8 the two diversity metrics (cf. section 3.2) are compared for different population sizes. The values listed there are the averaged mean values of the active population over all generations and all runs.

Table 8. Averaged diversity measures for classic and modified genetic algorithm

Population size (active+inactive)	**200**	**400**	**800**	**2000**
StandardDeviationOfFitness (classic)	0.00074	0.00075	0.00076	0.00076
StandardDeviationOfFitness (modified)	0.00198	0.00200	0.00204	0.00206
p-value	0	0	0	0
more diverse population	modified	modified	modified	modified
GenomeVariability (classic)	9.59	9.84	9.99	10.06
GenomeVariablility (modified)	9.45	9.82	10.05	10.18
p-value	0	0.043	0	0
more diverse population	classic	classic	modified	modified

The StandardDeviationOfFitness metric clearly indicates that exchanging individuals between populations increases the diversity in terms of fitness in a population. This finding is consistent with the GenomeVariability only for larger populations, though. Although hypothesis 2 cannot be confirmed generally by the results of this basic experiment, there is substantial evidence that migrating individuals increases the overall diversity in the active population.

Besides evaluating the diversity of the proper population it is also interesting to have a look how the *inactive population* is affected by the migration. Table 9 presents a number of averaged metrics for the inactive population of the experiment with total population size of 800 (migration rate 2, migration amount 3%). In this case the diversity of the inactive population is increased by the immigrating individuals. This is due to the random selection of immigrants from the proper population. In a scenario in which more individuals migrate (third row in the table) the inactive population loses its ability to maintain diversity. Instead, it tracks the fitness function almost as precisely as the active population does (for comparison: the mean fitness of the active population is 0.74 in this run). This finding can explain the poor performance for scenarios with a low migration interval but a high migration amount: In an event of change the migrated individuals are not distinct enough and thus cannot confer adaptive advantage any more.

Table 9. Averaged diversity measures for inactive population (total size 800)

Inactive population	**mean fitness**	**StandardDeviationOfFitness**	**GenomeVariability**
classic algorithm	0.50	0.00201	20.01
modified algorithm (amount=3%)	0.53	0.00523	22.59
modified algorithm (amount=20%)	0.65	0.00355	16.11

6 Discussion and future work

Naturally, it is difficult to generalise the results presented in this paper. They are only valid for the Zero-One-Max fitness function. The application of the two migration strategies to various other fitness functions may reveal a more complete picture. Furthermore, the fact that continuous migration performs poorly in context of a continuous fitness function is non-intuitive. While my results suggest that migration is the wrong approach for this sort of fitness functions, again this may well be problem-specific. More research in this direction seems necessary, therefore.

Another opportunity for future work is the use of different selection methods for determining the individuals to migrate and the ones to replace. In the presented experiments random selection was used in the active population, tournament selection was used in the inactive one. A preliminary experiment using *random selection* for both populations has shown that the selection method has a non-negligible influence: Randomly selected immigrants performed significantly worse ($\alpha = 0.05$). While this is a quite intuitive result, an in-depth look into different selection methods seems necessary.

A more profound evaluation of diversity in the subpopulations could be another area of future research. The presented averaged metrics can only serve as a vague indicator as they are not exploiting the existing problem knowledge. For a start, only the diversity of generations at which a change occurred could be evaluated. Furthermore, it would be especially interesting to evaluate diversity introduced by random individuals in order to understand their superior performance in experiment 1.

Finally, a change of the general setup is possible. The approach in this paper exchanges individuals between subpopulations completely independently in two steps. Another option might be to swap specific individuals between the subpopulations or swapping the whole populations. Based on this modified setup a comparison between swapping individuals and swapping parts of the genome like in the Swappers approach [6] would be possible.

7 Conclusion

This paper has shown that exchanging individuals between subpopulations can confer adaptive advantage. The most prominent performance increase was observed for abruptly changing fitness functions which were tackled with abrupt migration. In all other cases, though, the performance of the presented modification to the genetic algorithm was not able to offer superior performance. An important finding was the fact that applying the wrong exchanging strategy to a dynamic fitness function leads to extremely poor performance.

In terms of diversity the modified algorithm was more successful. The exchanged individuals increased the diversity in both, the active and the inactive population. Apart from these results, a number of performance evaluation methods and metrics were presented which may be of general use for further research.

References

1. H.G. Cobb. An investigation into the use of hypermutation as an adaptive operator in genetic algorithms having continuous, time-dependent nonstationary environments. Technical report, Naval Research Laboratory, Washington, 1990.
2. D.E. Goldberg and R.E. Smith. Nonstationary function optimization using genetic algorithms with dominance and diploidy. In J.J. Grefenstette, editor, *International Conference on Genetic Algorithms*, pages 59–68. Lawrence Erlbaum Associates, 1987.
3. J.H. Holland. *Adaptation in natural and artificial systems*. University of Michigan Press, 1975.
4. H. Homayounfar, S. Areibi, and F. Wang. An advanced island based GA for optimization problems. In *The Third International DCDIS Conference on Engineering Applications and Computational Algorithms*, 2003.
5. L.Z. Klinkmeijer. A serial population genetic algorithm for dynamic optimization problems. http://www.cs.uu.nl/˜dejong/publications/spa.pdf (visited on 2006-11-22), 2006.
6. J.R. Levenick. Swappers: Introns promote flexibility, diversity and invention. In Morgan Kaufmann, editor, *GECCO-99 Proceedings of the Genetic and Evolutionary Computation Conference*, pages pp. 361–368, 1999.
7. J. Lewis, E. Hart, and G. Ritchie. A comparison of dominance mechanisms and simple mutation on non-stationary problems. In A.E. Eiben, T. Baeck, M. Schoenauer, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature in LNCS*, volume 1498, pages 139–148. Springer, 1998.
8. M. Mitchell, S. Forrest, and J.H. Holland. The royal road for genetic algorithms: Fitness landscapes and GA performance. In *Proceedings of the First European Conference on Artificial Life*. MIT Press, 1991.
9. J.D. Schaffer and D. Offutt L.J. Eshelman. Spurious correlations and premature convergence in genetic algorithms. In G.J.E. Rawlins, editor, *Foundations of GAs and Classifier Systems*. Kaufmann, San Mateo, CA, 1991.