A Comparative Study of Evolutionary Optimization Techniques in Dynamic Environments

Demet Ayvaz
Computer Engineering Dept.
Bogazici University
Bebek, Istanbul, Turkey
demet.ayvaz@boun.edu.tr

Haluk Topcuoglu
Computer Engineering Dept.
Marmara University
Goztepe, Istanbul, Turkey
haluk@eng.marmara.edu.tr

Fikret Gurgen
Computer Engineering Dept.
Bogazici University
Bebek, Istanbul, Turkey
gurgen@boun.edu.tr

ABSTRACT
Genetic Algorithms have widely been used for solving optimization problems in stationary environments. In recent years, there has been a growing interest for investigating and improving the performance of these algorithms in dynamic environments where the fitness landscape changes. In this study, we present an extensive comparison of several algorithms with different characteristics on a common platform by using the moving peaks benchmark and by varying problem parameters.

Categories and Subject Descriptors
I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods and Search – *Heuristic Methods*
G.1.6 [**Numerical Analysis**]: Optimization – *global optimization*

General Terms
Performance, Experimentation

Keywords
Genetic Algorithms, Dynamic Environments, Multimodal Optimization, Performance Evaluation.

1. INTRODUCTION
In stationary optimization problems, it is assumed that the fitness landscape does not change during optimization process. In such an environment the goal of an optimization algorithm is to locate a stationary optimum. For real world optimization problems, the environment is usually dynamic and the goal of an optimization algorithm is to locate a changing optimum continually. Such an algorithm should have an adaptive behavior.

A set of evolutionary optimization techniques in dynamic environments have been proposed over the past few years, which have different design philosophies and characteristics. Additionally, researchers consider a set of syntactically-generated benchmarks [2] as well as benchmarks from real world problems [2,6] to measure the performance of their methods. These problems are not the same and the characteristics of the changes in an environment may generate different dynamic environments.

GECCO'06, July 8–12, 2006, Seattle, Washington, USA.
ACM 1-59593-186-4/06/0007.

The aim of this paper is to present a complete and an extensive performance evaluation of leading evolutionary optimization techniques in dynamic environments. We have examined and implemented a set of 13 evolutionary optimization techniques on a common platform and tested them using the same suite of benchmarks with a wide range of parameters.

2. A CLASSIFICATION of RELATED WORK
This section provides a classification of various techniques that deal with dynamic environments. It should be noted that it is not the purpose of this paper to provide a survey of all techniques; instead, it includes various leading techniques from each category. Evolutionary optimization techniques can be broadly classified into three categories [1] according to their functionalities: i) approaches that maintain diversity throughout the run, ii) memory-based approaches, and iii) multi-population approaches. Additionally a forth category is also considered in the classification in order to represent hybrid methods.

The simple evolutionary algorithm (SEA), the random immigrants (RI) [2], the evolutionary algorithm with three sub-populations (P3) [2] and the niching method[5] are the techniques considered in the category of maintaining the diversity throughout the run. Memory-based approaches [2] considered in our experimental study are the simple evolutionary algorithm with memory (SEAm), random immigrants with memory (RIm), evolutionary algorithm with three sub-populations and memory (P3m), memory/search method and the memory/2search method. The multi-population algorithms considered in our experiments are the self-organizing scouts method (SOS) [2] and the multi-national GA [4].

Additionally, we include two extensions of the leading methods in our experiments, which can be considered under the "hybrid methods" category. The first method hybridizes the self organizing scouts (SOS) with local search technique by providing crossover hill-climbing [3] operator in place of simulated binary crossover operator. In this approach we support the scout population with a local search algorithm to increase its exploitation capability, which has been reduces as a result of small population sizes in scouts. The second hybrid method combines the multi-national-GA with random immigrants method.

3. EXPERIMENTAL STUDY
Moving peaks benchmark is considered for performance evaluation of the algorithms, which is a multi-modal landscape consisting of peaks with changing heights, widths and locations in

an *n* dimensional search space. In this work, we used the problem generator suggested by Branke [2].

Performance evaluation of algorithms is based on two main metrics: offline error and running time. Offline error is the running average of the difference between the best individual encountered so far and the optimum at any time. Some of the algorithms included in this work require different amount of computational effort. In order to take these computational cost differences into account, we report running times of algorithms.

Number of fitness function evaluations may also be used as a measure of computational effort. But this can be misleading, if an evolutionary algorithm uses hidden labour. Two algorithms using equal amount of fitness evaluations can have much different computational costs as a result of hidden labour.

3.1 Results and Discussion

Selected parameters of the moving peaks benchmark are varied to provide different test cases or experiments. Our first experiment presents the effect of "severity of change" on the performance of different algorithms. Shift length is the norm of the vectors that will be added to peak location. If the shift length is taken to be 1, this means that each peak moves by one unit at each change.

Table 1 shows the offline errors and execution times of algorithms for different shift lengths. The experiments are performed on a cluster of Pentium IVs with 1.6 GHz and 512 Mb memory, running Linux operating system.

Table 1: The offline error and the average execution time of the algorithms for different shift lengths

Algorithm	Shift Length					
	Offline Error			Execution Time		
	1.0	2.0	3.0	1.0	2.0	3.0
SEA	17,008	18,154	18,319	1,85	2,00	1,95
RI	11,070	11,890	12,058	2,20	2,35	2,20
P3	11,945	12,744	14,056	1,75	1,80	1,70
Niching	6,334	7,179	7,308	11,05	11,30	10,70
SEAm	17,236	18,200	18,714	1,65	1,70	1,65
RIm	13,193	13,626	14,008	2,00	2,15	2,05
P3m	12,398	13,196	14,911	1,55	1,60	1,55
Mem/Search	7,383	8,785	9,789	1,60	1,65	1,60
Mem/2Search	7,492	9,075	10,950	1,55	1,60	1,55
SOS	4,340	5,042	5,908	2,45	2,60	2,50
Multi-national	5,973	5,894	5,949	8,10	8,60	8,05
SOS+LS	**3,413**	**4,091**	**4,488**	8,15	8,20	7,80
MN+RI	4,337	5,052	5,242	8,50	8,75	8,25

As shift length increases, the performance of all algorithms degrades and the offline errors increase. All approaches which use a simple memory performs worse than their standard approaches without memory and the difference increases as shift length increases. That is because while peaks move, the memory becomes more useless. Mem/Search outperforms all other algorithms except the multi-national ones and the niching algorithm. But as the environment changes, memory/search is affected more than the others since memory becomes less useful.

Multi-population approaches are much more efficient than others and they are less affected from the change in severity since they are capable of detecting and tracking multiple optima. The least affected algorithm is the multi-national GA, which is for using the hill-valley detection algorithm. This clustering algorithm is able to predict locations of peaks and valleys even if they move severely. SOS+LS gives the best results since it increases the search capacity of scout populations with a local search technique.

Memory supported approaches work faster than the versions without memory, since memory is separated from the main population. Even though niching method makes less number of evaluations than SOS+LS and MN+RI, it has the highest time overhead since it uses crowding selection and the WAMS replacement.

The second experimentation presents the effect of change predictability by changing the value of λ, where λ is the correlation coefficient. All multi-population approaches are nearly not affected and all diversity maintaining algorithms are slightly affected from this change. Only small changes occur when compared to memory based approaches, which is due to the memorization process. If peaks move around their initial conditions, the memorized solutions become more useful. Otherwise peaks move along and the old solutions in memory become useless and degrade the performance of algorithms.

4. REFERENCES

[1] J. Branke. Evolutionary Algorithms for dynamic optimization problems – a survey. Technical Report 387, Institute AIFB, University of Kalsruhe, February 1999.

[2] J. Branke. Evolutionary Optimization in Dynamic Environments. Kluwer, 2001.

[3] M. Lozano, F. Herrera, N. Krasnogor, and D. Molina. Real-coded memetic algorithms with crossover hill-climbing. *Evolutionary Computation*, vol. 12, no. 3, pp. 273–302, 2004

[4] Rasmus K. Ursem. Multi-national GAs: Multimodal optimization techniques in dynamic environments. Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2000), pages 19–26, Las Vegas, Nevada, USA, 10-12 2000. Morgan Kaufmann.

[5] W. Cedeno and V. R. Vemuri. On the use of niching for dynamic landscapes. In Intl. Conf. on Evolutionary Computation, IEEE, 1997.

[6] S. C. Lin, E. D. Goodman and W.F. Punch. A genetic algorithm approach to dynamic job shop scheduling problems. *Seventh International Conference on Genetic Algorithms,* pages 481-488, 1997.