PARALLEL AND DISTRIBUTED EVOLUTIONARY COMPUTATIONS FOR MULTIMODAL FUNCTION OPTIMIZATION*

VARUN RUPELA and GERRY DOZIER
Department of Computer Science and Software Engineering
107 Dunstan Hall, Auburn University, Alabama 36849-5347
E-mail: {vrupela/gvdozier}@eng.auburn.edu

ABSTRACT

A number of Evolutionary Computations (ECs) have been developed for solving Multimodal Function Optimization Problems (MFOPs) [1]. Some of the well-known ones are: Fitness Sharing [2], Sequential Niching [3], Simple Subpopulation Schemes [4] and Co-evolutionary Shared Niching [5]. These ECs have shown the capability of solving MFOPs, but have introduced one or more parameters that cannot be easily set without prior knowledge of the fitness landscape. Moreover, *a priori* knowledge of a particular MFOP may not always be readily available. In this work, we describe a set of parallel and distributed ECs that are capable of locating all the peaks in a MFOP without using parameters that require specific topological information. This paper also provides a performance comparison between three approaches to solving MFOPs: Fitness Sharing, parallel EC and distributed EC.

KEYWORDS multimodal, function optimization, parallel and distributed evolutionary computation, hill climbers.

INTRODUCTION

Evolutionary Computations (ECs) are optimization methods that have been praised for their ease of use and applicability on a wide range of problems without requiring any prior problem specific knowledge [6]. Although this may be the case for ECs designed to solve some conventional problems, this is not entirely true when it comes to the currently available ECs for solving Multimodal Function Optimization Problems (MFOPs) [1,2,3,4,5]. These ECs have used at least one parameter that is not easily set without the knowledge of the number of peaks and/or the distribution of the peaks in the MFOP. For example:

1) Fitness sharing [2] introduces the parameter, σ_{share}, which represents the maximum sharing distance.
2) Sequential niching [3] uses the parameter, r, which is referred to as the niche radius.
3) Co-evolutionary shared niching [4] introduces two new parameters, n_b and $d_{\min}$, where n_b denotes the number of businessmen and $d_{\min}$ denotes the minimum distance between the businessmen.
4) The Simple subpopulation scheme [5] introduces a maximum number of subpopulations parameter denoted by 2^n, where n is the number of tag bits.

* This research was supported by the National Science Foundation under grant #IIS-9907377.

Without the information required to set the parameters the user is left to a trial and error approach of modifying the parameters until a satisfactory result is obtained. Thus fixing the parameters prior to the execution of the EC without any topological information is a difficult task. The ECs introduced in this paper solve MFOPs in the absence of specific topological information about the search space.

The next section introduces a simple parallel EC and a concept of kings and dynasties that have been used to describe the two novel ECs: pEC2 and dEC1. The procedure used for data collection, the evaluation of the ECs, a performance comparison with Fitness Sharing and conclusions form the remaining sections of this paper.

A SIMPLE PARALLEL EVOLUTIONARY COMPUTATION

Consider an evolutionary computation that begins by randomly generating k candidate solutions (or individuals) in the search space. Each individual (is used as a parent and it) generates λ offspring by means of gaussian mutation[Ψ]. The best offspring of each parent is chosen to form a new generation of k individuals. This process continues until a stopping criterion is met. This EC is similar to k $(1,\lambda)$-Evolutionary Strategies running in parallel and independently to explore the search space. This EC is referred to as $(k,1,\lambda)$-pEC. For the sake of differentiating it from the ECs developed in this work, it will be referred to as pEC1.

Concept of Kingdoms and Dynasties

pEC1 can be considered to be working as follows: It begins by generating k individuals referred to as *kings* in the search space for exploring k regions[+]. Each *king* uses a radius σ to generate λ offspring. The best offspring of a *king* replaces it as the new *king*. This process of generating offspring and selecting new kings continues until the stopping criterion is met. The strategy thus forms k sets of hereditary rulers. In other words k *dynasties* are formed. Each dynasty can be considered to be an $(1,\lambda)$-ES that runs concurrently and independently to explore k regions of the search space.

The ECs discussed in the following sections are extensions of pEC1. They differ from it in the selection strategy used to form a new generation.

PARALLEL EVOLUTIONARY COMPUTATION 2 (pEC2)

Our first EC, which we will refer to as pEC2, works as follows:

1) Each *dynasty* keeps track of, $best_d$, the best individual found.
2) The offspring of all the *dynasties* are evaluated. A *dynasty* is considered *promising* if at least one of its current offspring is better than the *dynasty's* $best_d$.
3) In each *promising dynasty* the best offspring is made the new king.
4) Each *non-promising dynasty* is discontinued.

[Ψ] In gaussian mutation, the offspring is generated by adding a normally distributed random number $N(i,\sigma)$ to the individual i. σ represents the standard deviation of the distribution and is set prior to the execution of the EC.

[+] A region can be thought of as a small part of the search space.

5) For each discontinued *dynasty* a new *king* is randomly generated to begin a new *dynasty*.
6) The *kings* replacing a dynasty are evaluated when introduced.
7) The above steps are repeated until the stopping criterion is met.

The idea behind pEC2 is that once the best candidate solution of a region has been found the search must move to another region of the search space. This may save on the number of evaluations and also allow search in other presumably[♦] unexplored regions of the search space.

The above process assumes the non-improvement in the $best_d$ as an indication of the discovery of the best candidate solution of a region. But, as all possible candidate solutions of a region are not being evaluated, one non-improving generation cannot possibly guarantee the discovery of the best solution. Thus, in the event of a non-improving generation there may still be candidate solutions that are better fit than the $best_d$. Therefore, it is possible for pEC2 to prematurely discard a dynasty and cease exploration in a region. The next EC is based on providing a mechanism to reduce the chance of throwing away a region before finding the best candidate solution contained in it.

DISTRIBUTED EVOLUTIONARY COMPUTATION 1 (dEC1)

In addition to the (k,l,λ)-pEC above, we also developed a (k,l,λ)-dEC, which we refer to as dEC1. Unlike pEC2, each dynasty in dEC1 shares information about the fitness of its offspring with every other dynasty. This EC works as follows:

1) Each *dynasty* keeps track of $best_d$, the best individual found. The offspring of all the dynasties are evaluated and compared and the best k offspring saved as *best_set*.
2) A *dynasty* is considered *promising* if at least one of its current offspring is better than the *dynasty's* $best_d$. A *dynasty* is considered *contributing* if it has at least one of its offspring in the *best_set*.
3) The best offspring is made the new king for each *promising dynasty* and for each *non-promising* but *contributing dynasty*.
4) Each *non-promising and non-contributing dynasty* is discontinued.
5) For each discontinued *dynasty* a new *king* is randomly generated to begin a new *dynasty*.
6) The *kings* replacing a dynasty are evaluated when introduced.
7) The above steps are repeated until the stopping criterion is met.

A *contributing dynasty* is considered to be exploring a region that is better than most of the other regions. The above process prolongs exploration in a region by a *contributing dynasty*, even if it shows no improvement in the $best_d$. This gives the dynasty a chance to ensure that the best solution in a region is found before that region is discarded.

Following are some noteworthy features of pEC2 and dEC1:

[♦] We say 'presumably', due to the procedure used to replace a redundant dynasty. In the procedure used, a new candidate solution is selected at random from anywhere in the search space to be a king and begin a new dynasty. This procedure does not guarantee that the new candidate solution is from an unexplored region.

1) The number of evaluations made per generation varies with each generation and may be greater than or equal to $k*\lambda$. This is due to the evaluation of newly introduced *kings* that replace a discontinued dynasty.
2) The ECs utilize the effort spent on the search in the previous generations while selecting a new generation of individuals. This is true, as each dynasty's $best_d$ is used to determine if the dynasty is going to be retained for a new generation of search.
3) The ECs may be looked upon as a society of hill climbers [8], where a hill climber may be discontinued if it comes up with a non-improving generation.

TEST SUITE

The ECs were evaluated on the following test functions:

1. Goldberg's first function [5]:
   $f1(x) = \sin^6(5\pi x)$. $0 < x < 1$.
2. Goldberg's second function [5]:
   $f2(x) = e^{-2\log(2)\left(\frac{x-0.1}{0.8}\right)^2} \sin^6(5\pi x)$. $0 < x < 1$.
3. Himmelblau's Function [3]:
   $f3(x,y) = 200 - (x^2 + y - 11)^2 - (x + y^2 - 7)^2$. $-6 < x,y < 6$.

The functions f1(x) and f2(x) have 5 peaks each. The peaks are evenly distributed in the search space. While all the peaks of f1(x) are of the same height, peaks of f2(x) are not equal in height. The function f3(x,y) has 4 peaks of the same height that are unevenly distributed in the search space.

RESULTS

Parameters and Their Domains

The pEC2 and dEC1 evolutionary computations have the parameters k, λ and σ. GA with Fitness Sharing has the parameters μ, P_c and σ_{share} which refer to population size, probability of crossover, and maximum sharing distance respectively. Table I and II show the domains of the parameters that were used for the test functions.

Table I. List of Parameters and Their Domains for f1(x) and f2(x).

Parameters	Values						
μ, k	2	4	6	10	26	50	80
λ	1	3	5	10	15	20	25
σ	0.0005	0.00075	0.001	0.0025	0.005	0.0075	0.01
Pc	0.6	0.7	0.8	0.85	0.9	0.95	0.95
σ_{share}	0.025	0.05	0.075	0.1	0.15	0.2	0.25

Table II. List of Parameters and Their Domains for f3(x).

Parameters	Values						
μ, k	2	4	6	10	26	50	80
λ	1	3	5	10	15	20	25
σ	0.005	0.01	0.04	0.07	0.1	0.15	0.2

Pc	0.6	0.7	0.8	0.85	0.9	0.95	0.95
σ_{share}	2	3	4	5	6	7	8

Note that the 3rd row in Tables I and II provide the values for λ', which is the number of offspring generated by a single dynasty in one generation. λ is equal to $k * \lambda'$

Details of GA with Fitness Sharing

The GA Sharing used in this work is the same as the one used by Goldberg [2]. The GA uses a binary coded representation, stochastic remainder selection, single point crossover, no mutation and fitness sharing. Goldberg used a population size of 50, crossover rate of 0.8 and 100 generations as the stopping criterion for test functions f1(x) and f2(x). These values have been incorporated into the domain values shown in Tables I and II. The GA sharing implemented here used Goldberg's stopping criterion only with a population size of 50. The GA sharing used in this work used a bit string of length 32 while Goldberg used a bit string of length 30.

Procedure used for Data Collection

The following procedure was adopted for evaluating the ECs on each test function:

1) *For each test functions, each EC was executed 30 times on all of the possible parameter settings that can be formed from Tables I and II.* From Tables I and II, one can see that a total of $(7)^3 = 343$ parameter settings would be formed for each EC.
2) Each run was terminated when the number of evaluations exceeded a pre-determined *number of function evaluations*. This limit was set to 5000 for f1(x) and f2(x) and 10000 for f3(x,y).
3) If pEC2 and dEC1 found a candidate solution within $\pm w$ units of a peak then that peak was considered found. w was set to 0.00005 for f1(x) and f2(x) and to 0.01 for f3(x,y).[∇] Since GAs are not good at locating an optima precisely [7], for GA with Sharing, a peak was considered found if a cluster of individuals was formed around the peak. One or more individuals with fitness greater than or equal to 99% of the peaks fitness were considered to be a cluster.

Data Collected with each Test Function: For each of the 343 parameter settings that the ECs were subjected to, the following data was collected: (1) Mean number of peaks found in the 30 runs. We refer to this value as *MP*. (2) The number of runs out of 30 in which all the peaks of the test function were found. We refer to this value as *Runs/30*.

Performance Comparison and Conclusion

Table III shows a comparison of the performance of the ECs on the three test functions. The column "Best (Runs/30)" shows the best value for *Runs/30* obtained by each EC with respect to each Test Function. The column "% PS" shows the percentage of parameter settings with which the ECs obtained a value for *MP* that was greater than or equal to 90% of the total number of peaks, with respect to each test function. The following *conclusions* may thus be drawn with respect to the experiments conducted:

[∇] These values were chosen such that they do not allow random search to locate all of the peaks within the function evaluations limit.

1) pEC2 and dEC1 are capable of solving MFOPs without knowledge of the number and distribution of the peaks of the test functions.
2) Fitness Sharing clearly performs better than dEC1 and pEC1 on f2(x) in terms of the "Best (Runs/30)" and the "% PS criteria". Fitness sharing and dEC1 are comparable on f1(x). We point out that the peaks in f1(x) and f2(x) are evenly distributed. We speculate that dEC1 performs poorly on f2(x), a function with peaks of unequal height, as a dynasty exploring a region close to a low peak would never contribute to the search and will be discarded before it finds the peak. The peaks in f3(x,y) are unevenly distributed. Fitness sharing performs poorly on it.
3) During the experiments it was also noted that pEC2 and dEC1 were capable of solving the MFOPs with any k in the domain, while Fitness Sharing on a GA required the population size to be larger than the number of peaks in the MFOP.

Table III. Performance Comparison of pEC2, dEC1 and Fitness Sharing.

Test Functions	pEC2		dEC1		Fitness Sharing	
	Best (Runs/30)	% of PS	Best (Runs/30)	% of PS	Best (Runs/30)	% of PS
f1(x)	20	3.79	29	34.11	30	28.86
f2(x)	17	1.74	17	0.0	30	28.57
f3(x,y)	26	6.99	30	23.90	21	2.0

FUTURE WORK

We would like to modify dEC1 in order to improve its performance on MFOPs with peaks of varying heights. We would also like to test pEC2 and dEC1 on global optimization problems and combinatorial optimization problems including distributed constraint satisfaction problems.

REFERENCES

1. G. Harik. "Finding multiple solutions in problems of bounded difficulty." IlliGal report no. 94002, University of Illinois, The Illinois Genetic Algorithms Laboratory, Urbana-Champaign, 1994.
2. D. E. Goldberg and J. Richardson. "Genetic algorithms with sharing for multimodal function optimization." In J.J. Grefenstette, editor, *Proceedings of the Second International Conference on Genetic Algorithms*, pages 41-49. 1987.
3. D. Beasley, D. R. Bull, and R. R. Martin. "A sequential niche technique for multimodal function optimization." *Evolutionary Computation*, 1(2): 101-125, 1993.
4. W. M. Spears. "Simple subpopulation schemes." In *Evolutionary Programming Conference*, pages 296--307. World Scientific, 1994.
5. D. E. Goldberg and Liwei Wang. "Adaptive Niching Via Coevolutionary Sharing." IlliGal report no. 97007, University of Illinois, The Illinois Genetic Algorithms Laboratory, Urbana-Champaign, 1997.
6. Melanie Mitchell, "An Introduction to Genetic Algorithms". The MIT Press, 1996.
7. Kenneth A. De Jong. Genetic algorithms are NOT function optimizers. In D. Whitley, editor, *Foundations of Genetic Algorithms 2*, San Mateo, CA, 1993.
8. M. Sebag and M. Shoenauer, "A Society of Hill-Climbers." In *The Proceedings of the 1997 International Conference on Evolutionary Computation*, pp. 319-324, IEEE Press.