Parameter-Less Evolutionary Search

Gregor Papa
Computer Systems Department
Jožef Stefan Institute, Ljubljana, Slovenia
gregor.papa@ijs.si

ABSTRACT

The paper presents the parameter-less implementation of an evolutionary-based search. It does not need any predefined control parameters values, which are usually used for genetic algorithms and similar techniques. Efficiency of the proposed algorithm was evaluated by CEC2006 benchmark functions and a real-world product optimization problem.

Categories and Subject Descriptors

I.2.8 [**ARTIFICIAL INTELLIGENCE**]: Problem Solving, Control Methods, and Search

General Terms

Algorithms

Keywords

Parameter-less, Evolution, Search

1. INTRODUCTION

The aim was to develop a quick algorithm that would solve any problem without a specialist intervention for setting the suitable control parameters [1], [2], [3]. Parameter-less GA in [3] used fixed parameters. Authors found some values of the selection rate and crossover probability that ensure robust behavior of the algorithm. The self-adaptive differential evolution in [1] varied the value of amplification factor and crossover parameter, while population size was never changed during the optimization.

The Parameter-Less Evolutionary Search (PLES), initially proposed in [5], is based on GA, but control parameters are not set in advance. They are set according to complexity of the problem and according to statistical properties of the solutions. The suitability and efficiency of the proposed algorithm were evaluated by the CEC2006 benchmark functions [4], and one industrial optimization problem [6], [7].

2. PLES

The main advantage of PLES over the basic GA is the fact that PLES can set the control parameters like population size, number of generations, probabilities of crossover and mutation by itself, during the optimization process. The values of parameters depend on the statistical behavior and convergence of the solutions. In PLES; elitism, selection, and crossover are implemented through forcing of better individuals, while mutation is split between forcing of better individuals and moving of individuals. The control parameters are never set in advance and are not constant.

2.1 Setup

For n independent variables the chromosome looks like the string of n values, while for n dependent variables n positions and the order in the chromosome would represent dependencies described in the input specification. In the second case the interdependent variables are placed together or closer in the chromosome.

2.2 Initialization

The population size ($PopSize$) is proportional to chromosome size, i.e., problem complexity

$$PopSize = n + log_{10}(n) + 2log_{10}(Range) \tag{1}$$

where n is the number of variables to be optimized and

$$Range = \sum_{i=1}^{n}(max_i - min_i)10^{decplc_i} \tag{2}$$

where max_i and min_i are the upper and the lower limit of the i-th variable, respectively, and $decplc_i$ is the number of decimal places (the resolution) of the i-th variable.

2.3 Stopping criterion

The number of generations depends on the convergence of the best solution found. Optimization is running while better solution is found every few generations. The $Limit$ (i.e., number of generations since the last improvement) for stopping the optimization process is defined as

$$Limit = 10log_{10}(PopSize) + log_{10}(Resting + 1) \tag{3}$$

where $Resting$ is the number of generations since the last improvement of the global best solution.

2.4 Force better solution

In every generation the worse solutions are replaced with better solutions. Here, each s_{i_j} (variable j of the solution i) is randomly moved up to 10% of the difference between the variable and the limit (upper or lower, regarding the move direction) of that variable.

GECCO'08, July 12–16, 2008, Atlanta, Georgia, USA.
ACM 978-1-60558-130-9/08/07.

2.5 Solution moving

Mutation is realized by moving of some positions in the chromosome according to statistical properties. Only the solutions that were not moved by "Force better" operator are handled here. The number (*Ratio*) of the positions in the chromosome to be moved is

$$Ratio = \tanh\left(1 - \frac{StDev_{i-1}}{StDev_{max}}\right) \times n \qquad (4)$$

where $StDev_{i-1}$ and $StDev_{max}$ are the standard deviation of the solution fitness of the previous generation, and the maximal standard deviation of all generations, respectively. The size of the move is

$$Move = \tanh\left(\left|\frac{s_{best_j} - s_{i_j}}{average_j - s_{i_j}}\right|\right) \qquad (5)$$

where s_{i_j} is the value of the parameter j of the current solution i, and s_{best_j} is the value of parameter j of the globally best solution, $average_j$ is the average value of the parameter j in the previous generation.

$$Range = s_{best_j} - s_{i_j} \qquad (6)$$

$$s_{i_j} = s_{i_j} + Direction \times Move \times Range \qquad (7)$$

where $Direction$ is randomly selected number $\{-1, 1\}$ to determine the direction of the move.

2.6 Statistical evaluation

Each population is statistically evaluated. Here the best, the worst, and average fitness value in the generation is found. Furthermore, the standard deviation of fitness values of all solutions in the generation, maximal standard deviation of fitness value over all generations, and average value of each parameter in the solution is calculated.

3. EVALUATION RESULTS

The first experiment for the evaluation of the PLES was performed by some CEC2006 benchmark functions defined for constrained real-parameter optimization [1], [4]. In the second experiment we optimized geometrical parameters of the electrical motor rotor and stator [6], [7].

Table 1: Results for CEC2006 functions: g01 – g03

	g01	g02	g03
Optimal	-15.000000	-0.803619	-1.000500
Best	-14,892690	-0,733780	-1,000350
Worst	-10,715800	-0,314170	-0,984590
Average	-13,586952	-0,604088	-0,998541
Mean	-14,042460	-0,641730	-0,999570
St.dev.	1,191597	0,105540	0,005256
Evaluations	11.336	19.996	10.037

The best, worst, average, and mean value of the solutions after 20 runs are presented in Table 1, Table 2, and Table 3. Further also standard deviation of solutions and the average number of evaluations is presented.

The results presented in Tables 1 and 2 show that PLES is able to come close to the optimal solution very quickly, even it the optimal solution is surrounded by unfeasible regions.

Table 2: Results for CEC2006 functions: g04 – g06

	g04	g05	g06
Optimal	-30.665.53867	5.126.49671	-6.961.81387
Best	-30.856,52391	5.134,20061	-6.970,11257
Worst	-30.817,80674	5.693,79575	-7.527,66061
Average	-30.836,55150	5.317,69385	-7.334,02541
Mean	-30.839,10520	5.272,10264	-7.507,03885
St.dev.	9,73687	153,69495	273,44339
Evaluations	15.082	6.245	4.811

Table 3: Results for motor geometry optimization

	PLES	GEA	MASA
Best	135.4	131.3	114.2
Worst	147.9	139.9	135.9
Average	141.1	132.9	128.9
Mean	140.2	–	–
St.dev.	3.7	3.3	7.8
Evaluations	1692	1400	1400

Comparing to [7] the PLES is able to find solutions as fast as the other GA-based approaches.

4. CONCLUSION

The presented algorithm does not need any predefined control parameter values, and no special knowledge is needed to effectively use the algorithm. The efficiency of the algorithm was evaluated and proved by CEC2006 benchmark functions and a real-world industrial problem.

5. REFERENCES

[1] J. Brest, S. Greiner, B. Bošković, M. Mernik, and V. Žumer. Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems. *IEEE Transactions on Evolutionary Computation*, 10(6):646–657, December 2006.

[2] J. Gomez. Self adaptation of operator rates in evolutionary algorithms. In *GECCO 2004*, pages 1162–1173, 2004.

[3] G. R. Harik and F. G. Lobo. A parameter-less genetic algorithm. In *GECCO 1999*, pages 258–265, July 1999.

[4] J. Liang, T. Runarsson, E. Mezura-Montes, M. Clerc, P. Suganthan, C. C. Coello, and K. Deb. Problem definitions and evaluation criteria for the cec 2006 special session on constrained real-parameter optimization. Technical Report 2006005, Nanyang Technological University, Singapore, March 2006.

[5] G. Papa. *Concurrent operation scheduling and unit allocation with an evolutionary technique in the process of integrated-circuit design*. PhD thesis, University of Ljubljana, Ljubljana, Slovenia, October 2002.

[6] G. Papa and B. Koroušić-Seljak. An artificial intelligence approach to the efficiency improvement of a universal motor. *Engineering Applications of Artificial Intelligence*, 18(1):47–55, February 2005.

[7] T. Tušar, P. Korošec, G. Papa, B. Filipič, and J. Šilc. A comparative study of stochastic optimization methods in electric motor design. *Applied Intelligence*, 27(2):101–111, October 2007.