Use of Domain Information to Improve the Performance of an Evolutionary Algorithm

Ricardo Landa Becerra
Evolutionary Computation Group (EVOCINV)
CINVESTAV-IPN, Computer Science Section,
Electrical Engineering Department
México, D.F., 07300, MEXICO
rlanda@computacion.cs.cinvestav.mx

Carlos A. Coello Coello
Evolutionary Computation Group (EVOCINV)
CINVESTAV-IPN, Computer Science Section
Electrical Engineering Department
México, D.F., 07300, MEXICO
ccoello@cs.cinvestav.mx

ABSTRACT

The main goal of this thesis work is to explore the capacities of cultural algorithms to add domain knowledge in evolutionary computation. Within our objectives is to develop a cultural algorithm for constrained optimization, and other for multiobjective optimization. With a proper desing of the belief space we expect to obtain competitive results compared with other state-of-the-art evolutionary algorithms, but reducing the number of fitness function evaluations needed. In this paper we focus in the algorithm for constrained optimization, because the development of the algorithm for multiobjective optimzation is an early stage.

Categories and Subject Descriptors

G.1.6 [**Numerical Analysis**]: Optimization—*Constrained optimization, Global optimization*; I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search—*Heuristic methods*

General Terms

Algorithms, design

Keywords

Constraint handling, optimization, cultural algorithms, differential evolution

1. INTRODUCTION

In evolutionary computation it is common that the only information available about the problem we are solving is the evaluation of the objective function at a given point. This is good when we need generality, because no other specific characteristics of a problem are involved in its performance. However, incorporating domain knowledge to evolutionary algoritmhs may improve their performance if we want to use them only in a certain class of problems (as constrained optimization problems we deal in this paper).

Sometimes we don't have the characteristics of the problem we want to solve before the execution of the algorithm. An alternative are cultural algorithms, that extract domain knowledge from the problem during the evolutionary process.

The remainder of this paper is organized as follows. In Section 2, we provide some basic concepts related to cultural algorithms. In Section 3, we describe the basic differential evolution algorithm. In Section 4, we describe our proposed approach for constrained optimization. In Section 5, we show some results of the algorithm for constrained optimization using a well-known benchmark. Finally, in Section 6, we draw our conclusions and ennumerate the work that remains to be done.

2. CULTURAL ALGORITHMS

Cultural algorithms are techniques that add domain knowledge to evolutionary computation methods. They are based on the assumption that domain knowledge can be extracted during the evolutionary process, by means of the evaluation of each point generated [6]. This process of extraction and use of the information, has been shown to be very effective in decreasing computational cost while approximating global optima, in unconstrained, constrained and dynamic optimization [2, 3, 8].

Cultural algorithms consist of two main components: the population space, and the belief space [6]. The **population space** consists of a set of possible solutions to the problem, and can be modeled using any population-based technique, e.g. genetic algorithms. The **belief space** is the information repository in which the individuals can store their experiences for the other individuals to learn them indirectly. In cultural algorithms, the information acquired by an individual can be shared with the entire population.

Both spaces (i.e., population space and belief space) are linked through a communication protocol, which states the rules about the individuals that can contribute to the belief space with their experiences (the acceptance function), and the way the belief space can influence to the new individuals (the influence function).

When designing a cultural algorithm, the first key factor is the popolation space. As we said before, any population-

GECCO'05, June 25–29, 2005, Washington, DC, USA.

```
Generate initial population
Evaluate initial population
Initialize the belief space
Do
    For each individual in the population
        Apply the variation operator influenced by a
            randomly chosen knowledge source
        Evaluate the child generated
        Replace the individual with the child, if the child
            is better
    End for
    Update the belief space with the accepted
        individuals
Until the termination condition is achieved
```

Figure 2: Pseudo-code of the cultured differential evolution.

based technique can be used as a population space, but the robustness of search engine may be very helpful in obtainig good results.

Differential evolution [5] is a recently developed evolutionary algorithm, focused on the solution of real paramenter optimization problems. Differential evolution has been found to be a very robust optimization technique [9]. However, this is the first proposal that uses differential evolution as the population space of a cultural algorithm.

3. DIFFERENTIAL EVOLUTION

Differential evolution is an evolutionary algorithm originally proposed by Price and Storn [5], whose main design emphasis is real parameter optimization. Differential evolution is based on a mutation operator, which adds an amount obtained by the difference of two randomly chosen individuals of the current population, in contrast to most of the evolutionary algorithms, in which the mutation operator is defined by a probability function.

The basic algorithm of differential evolution is shown in Figure 1, where the problem to be solved has n decision variables, F and CR are parameters given by the user, and $x_{i,j}$ is the i-th decision variable of the j-th individual in the population.

4. THE PROPOSED APPROACH FOR CONSTRAINED OPTIMIZATION

We have already developed a cultural algorithm for constrained optimization. Our proposed approach uses differential evolution in the population space. A pseudo-code of our approach (called cultured differential evolution) is shown in Figure 2.

In the initial steps of the algorithm, a population of *popsize* individuals is created, as well as a belief space. For the offspring generation, the variation operator of the differential evolution algorithm is influenced by the belief space.

Since we want to solve constrained optimization problems, the objective function by itself does not provide enough information as to guide the search properly. To determine if a child is better than its parent, and, therefore, it can replace it, we use the following rules:

1. A feasible individual is better than an infeasible one.

l_1	u_1	l_2	u_2	$\cdots$	l_n	u_n
L_1	U_1	L_2	U_2	$\cdots$	L_n	U_n

dm_1	dm_2	$\ldots$	dm_n

Figure 3: Structure of the normative knowledge

2. If both are feasible, the individual with the best objective function value is better.

3. If both are infeasible, the individual with less amount of constraint violations is better.

The amount of constraint violation is measured with normalized constraints, with the use of the following expression:

$$viol(x_j) = \sum_{c=1}^{constr} \frac{g_c(x_j)}{g_{maxc}}$$

where $g_c(x)$ are the *constr* constraints of the problem, and g_{maxc} is the largest amount of violation of the constraint $g_c(x)$ found so far.

4.1 The Belief Space

In our approach, the belief space is divided in four knowledge sources, described next.

4.1.1 Situational Knowledge

Situational knowledge consists of the best exemplar E found along the evolutionary process. It represents a leader for the other individuals to follow.

The variation operators of differential evolution are influenced in the following way:

$$x'_{i,j} = E_i + F * (x_{i,r1} - x_{i,r2})$$

where E_i is the i-th component of the individual stored in the situational knowledge. This way, we use the leader instead of a randomly chosen individual for the recombination. This has the effect of pushing the children closer to the best point found.

4.1.2 Normative Knowledge

The normative knowledge contains the intervals for the decision variables where good solutions have been found, in order to move new solutions towards those intervals. Thus, the normative knowledge has the structure shown in Figure 3.

In Figure 3, l_i and u_i are the lower and upper bounds, respectively, for the i-th decision variable, and L_i and U_i are the values of the fitness function associated with that bound. Also, the normative knowledge includes the values dm_i, to influence the mutation operator adopted in differential evolution. The values dm_i store the largest difference $|x_{i,r1} - x_{i,r2}|$ found during the application of the variation operators at the previous generation.

The following expression shows the influence of the normative knowledge on the variation operators:

$$x'_{i,j} = \begin{cases} x_{i,r3} + F * |x_{i,r1} - x_{i,r2}| & \text{if } x_{i,r3} < l_i \\ x_{i,r3} - F * |x_{i,r1} - x_{i,r2}| & \text{if } x_{i,r3} > u_i \\ x_{i,r3} + \frac{u_i - l_i}{dm_i} * F * (x_{i,r1} - x_{i,r2}) & \text{otherwise} \end{cases}$$

Generate initial population of size *popsize*
Do
  For each individual j in the population
    Generate three random integers, r_1, r_2 and $r_3 \in (1, popsize)$,
      with $r_1 \neq r_2 \neq r_3 \neq j$
    Generate a random integer $i_{rand} \in (1, n)$
    For each parameter i
      $x'_{i,j} = \begin{cases} x_{i,r3} + F * (x_{i,r1} - x_{i,r2}) & \text{if } rand(0,1) < CR \text{ or } i = i_{rand} \\ x_{i,j} & \text{otherwise} \end{cases}$
    End For
    Replace x_j with the child x'_j, if x'_j is better
  End For
Until the termination condition is achieved

Figure 1: Pseudo-code of the differential evolution algorithm adopted in this work (this version is called DE/rand/1/bin)

e_1	$\cdots$	e_i	$\cdots$	e_w

ds_1	ds_2	$\cdots$	ds_n
dr_1	dr_2	$\cdots$	dr_n

Figure 4: Structure of the history knowledge

We introduce the scaling factor $\frac{u_i - l_i}{dm_i}$ for the mutation to be proportional to the interval of the normative knowledge for the i-th decision variable.

4.1.3 Topographical Knowledge

The usefulness of the topographical knowledge is to create a map of the fitness landscape of the problem during the evolutionary process. It consists of a set of cells, and the best individual found on each cell. The topographical knowledge, also, has an ordered list of the best b cells, based on the fitness value of the best individual on each of them. For the sake of a more efficient memory management, in the presence of high dimensionality (i.e., too many decision variables), we use an spatial data structure, called k-d tree, or k-dimensional binary tree [1]. In k-d trees, each node can only have two children (or none, if it is a leaf node), and represents a division in half for any of the k dimensions.

The influence function tries to move the children to any of the b cells in the list:

$$x'_{i,j} = \begin{cases} x_{i,r3} + F * |x_{i,r1} - x_{i,r2}| & \text{if } x_{i,r3} < l_{i,c} \\ x_{i,r3} - F * |x_{i,r1} - x_{i,r2}| & \text{if } x_{i,r3} > u_{i,c} \\ x_{i,r3} + F * (x_{i,r1} - x_{i,r2}) & \text{otherwise} \end{cases}$$

where $l_{i,c}$ and $u_{i,c}$ are the lower and upper bounds of the cell c, randomly chosen from the list of the b best cells.

4.1.4 History Knowledge

This knowledge source was originally proposed for dynamic objective functions, and it was used to find patterns in the environmental changes [8]. History knowledge records in a list, the location of the best individual found before each environmental change. That list has a maximum size w.

The structure of the history knowledge is shown in Figure 4, where e_i is the best individual found before the i-th environmental change, ds_i is the average distance of the changes for parameter i, and dr_i is the average direction if there are changes for parameter i. In our approach, instead of detecting changes of the environment, we store a solution if it remains as the best one during the last p generations. If this happens, we assume that we are trapped in a local optimum.

The expression of the influence function of the history knowledge is the following:

$$x'_{i,j} = \begin{cases} e_{i,1} + dr_i * F * |x_{i,r1} - x_{i,r2}| & \text{if } rand(0,1) < \alpha \\ e_{i,1} + \frac{ds_i}{dm_i} * (x_{i,r1} - x_{i,r2}) & \text{if } rand(0,1) < \beta \\ rand(lb_i, ub_i) & \text{otherwise} \end{cases}$$

where $e_{i,1}$ is the i-th decision variable of the previous best e_1 stored in the list of the history knowledge, dm_i is the maximum difference for the i-th variable, stored in the normative knowledge, lb_i and ub_i are the lower and upper bounds of the variable x_i, given as input for the problem, and the function $rand(a, b)$ returns a random number between its two arguments, say a and b.

4.2 Main Influence Function

The main influence function is responsible for choosing the knowledge source to be applied to the variation operator of differential evolution, based on the success rate of knowledge sources at the given generation. This way, the use of different knowledge sources responds to the dynamics of the evolutionary process.

At the beginning, all the knowledge sources have the same probability to be applied, $\%p_{ks} = \frac{1}{4}$, because there are 4 knowledge sources; but during the evolutionary process, the probability of the knowledge source ks to be applied is:

$$\%p_{ks} = 0.1 + 0.6\frac{v_{ks}}{v}$$

where v_{ks} are the times that an individual generated by the knowledge source ks outperforms its parent in the current generation, and v are the times that an individual generated (by any knowledge source) outperforms its parent in the current generation. The lower bound of $\%p$ is the arbitrary value 0.1, to ensure that any knowledge source has always a probability > 0 to be applied. If $v = 0$ during a generation, $\%p_{ks} = \frac{1}{4}$, as in the beginning.

5. SOME RESULTS

To validate our approach, we adopted the well-known benchmark originally proposed in [4] and extended in [7] which has been often used in the literature to validate new

TF	Optimal	Best		Mean		Worst		St. Dev.	
		CDE	SR	CDE	SR	CDE	SR	CDE	SR
g01	-15	**-15.000000**	**-15.000**	-14.999996	**-15.000**	-14.999993	**-15.000**	0.000002	0.0
g02	0.803619	**0.803619**	0.803515	0.724886	0.781975	0.590908	0.726288	0.070125	0.020
g03	1	0.995413	**1.000**	0.788635	**1.000**	0.639920	**1.000**	0.115214	0.00019
g04	-30665.539	**-30665.539**	**-30665.539**	**-30665.539**	**-30665.539**	**-30665.539**	**-30665.539**	0.000000	0.00002
g05	5126.4981	5126.570923	5126.497	5207.410651	5128.881	5327.390497	5142.472	69.225796	3.5
g06	-6961.8138	**-6961.8139**	**-6961.814**	**-6961.8139**	-6875.940	**-6961.8139**	-6350.262	0.000000	160
g07	24.3062091	**24.306209**	24.307	24.306210	24.374	24.306212	24.642	0.000001	0.066
g08	0.095825	**0.095825**	**0.095825**	**0.095825**	**0.095825**	**0.095825**	**0.095825**	0.000000	0.000000
g09	680.630057	**680.630057**	**680.630**	**680.630057**	680.656	**680.630057**	680.763	0.000000	0.034
g10	7049.25	**7049.2481**	7054.316	**7049.2483**	7559.192	**7049.2485**	8835.655	0.000167	530
g11	0.75	**0.749900**	**0.750**	0.757995	**0.750**	0.796455	**0.750**	0.017138	0.00008
g12	1	**1.000000**	**1.000000**	**1.000000**	**1.000000**	**1.000000**	**1.000000**	0.000000	0.0
g13	0.0539498	0.056180	0.053957	0.288324	0.067543	0.392100	0.216915	0.167095	0.031

Table 1: Comparison of the results obtained by our cultured differential evolution approach (CDE), and the stochastic ranking approach (SR)

constraint-handling techniques. For a further description of those problems, refer to [7]. We compare against the most competitive evolutionary algorithm for constrained optimization: the stochastic ranking approach [7].

The results shown in Table 1 were obtained by the cultured differential evolution only with **100,100** evaluations of the fitness function, whereas the stochastic ranking obtained its results with 350,000 evaluations. We can see that the approximations of the optimal values obtained by the cultured differential evolution are very competitive in most of the cases, and are obtained with a low number of fitness function evaluations. This can be attributed to a faster exploration of the influenced operators by the belief space. The robustness is a feature of the search engine (i.e. differential evolution), also increased by the knowledge sources that allows exploitation, such as the situational knowledge.

6. CONCLUSIONS AND FUTURE WORK

Until now, we have developed a cultural algorithm for constrained optimization. With it, we obtain competitive results with a considerably lower number of fitness function evaluations. In addition to the standard benchmark, we carried out some more experiments, that will be published soon.

We are currently developing the cultural algorithm for multiobjective optimization that we projected since the beggining of the work. We expect to have similar performance to the algorithm for constrained optimization. In order to verify this hypothesis, it is necessary to make some experiments with a proper benchmark, and compare the results with other state-of-the-art algorithms.

Acknowledgements

The first author acknowledges support from CONACyT to pursue graduate studies at the Computer Science Section at CINVESTAV-IPN. The second author gratefully acknowledges support from CONACyT through project 42435-Y.

7. REFERENCES

[1] J. L. Bentley and J. H. Friedman. Data Structures for Range Searching. *ACM Computing Surveys*, 11(4):397–409, December 1979.

[2] C.-J. Chung and R. G. Reynolds. CAEP: An Evolution-based Tool for Real-Valued Function Optimization using Cultural Algorithms. *Journal on Artificial Intelligence Tools*, 7(3):239–292, 1998.

[3] X. Jin and R. G. Reynolds. Using Knowledge-Based Evolutionary Computation to Solve Nonlinear Constraint Optimization Problems: a Cultural Algorithm Approach. In *1999 Congress on Evolutionary Computation*, pages 1672–1678, Washington, D.C., July 1999. IEEE Service Center.

[4] Z. Michalewicz and M. Schoenauer. Evolutionary Algorithms for Constrained Parameter Optimization Problems. *Evolutionary Computation*, 4(1):1–32, 1996.

[5] K. V. Price. An introduction to differential evolution. In D. Corne, M. Dorigo, and F. Glover, editors, *New Ideas in Optimization*, pages 79–108. McGraw-Hill, London, UK, 1999.

[6] R. G. Reynolds. Cultural algorithms: Theory and applications. In D. Corne, M. Dorigo, and F. Glover, editors, *New Ideas in Optimization*, pages 367–377. McGraw-Hill, London, UK, 1999.

[7] T. P. Runarsson and X. Yao. Stochastic Ranking for Constrained Evolutionary Optimization. *IEEE Transactions on Evolutionary Computation*, 4(3):284–294, September 2000.

[8] S. M. Saleem. *Knowledge-Based Solution to Dynamic Optimization Problems using Cultural Algorithms*. PhD thesis, Wayne State University, Detroit, Michigan, 2001.

[9] R. Storn. System Design by Constraint Adaptation and Differential Evolution. *IEEE Transactions on Evolutionary Computation*, 3(1):22–34, April 1999.