

Paper:

# A Comparison of Differential Evolution and Generalized Generation Gap Model

Jani Rönkkönen, Saku Kukkonen, and Jouni Lampinen

Laboratory of Information Processing, Department of Information Technology, Lappeenranta University of Technology

P.O. Box 20, FIN-53851, Lappeenranta, Finland

E-mail: {jani.ronkkonen, saku.kukkonen, jouni.lampinen}@lut.fi

[Received December 4, 2004; accepted May 19, 2005]

**We compared two floating-point-encoded evolutionary algorithms (EA) – differential evolution (DE) and the generalized generation gap (G3) – using a set of problems with different characteristics. G3 is reported to offer superior performance with unimodal functions, which are, however, often solved more efficiently using derivative-based optimization for example and it is interesting to know, how these algorithms perform in multimodal global optimization problems.**

**Our results suggest that G3 converges fast but is prone to converge prematurely rather than finding the global optimum in high-dimensional multimodal problems. DE, in contrast, appears to handle multimodal problems better but cannot match convergence speed of G3 in unimodal problems.**

**Keywords:** differential evolution, global optimization, generalized generation gap model, parent-centric recombination

## 1. Introduction

A new real-parameter genetic algorithm called the generalized generation gap (G3) using the generic parent-centric recombination (PCX) operator has been proposed [4]. The PCX operator has been shown to outperform previous unimodal normal distribution crossover and simplex crossover operators. Empirical results from a comparison between the G3 algorithm and other optimization algorithms, including differential evolution (DE) algorithm, suggest that G3 has performance superior to DE. The test setup was, however, rather limited, including only three differentiable test functions, two of which were unimodal. Real-world optimization problems are often high-dimensional, multimodal, and nonseparable, and good performance in a unimodal case does not necessarily guarantee good performance in more complex problems. The effect of different control parameter setups was not thoroughly studied, either, leaving a question as to whether the results can be generalized to more demanding multimodal functions and to what extent this is possible.

We have added to the earlier investigation [4] by making a more comprehensive comparison between G3 and

DE algorithms using 12 different test functions in convergence speed and the algorithm's ability to reach the global optimum. We looked for a good control parameter setup for each problem to make the comparison as objective as possible and to determine the real potential of the algorithms rather than fixing control parameters and favoring one or the other with a more suitable control parameter setup.

Our objective was to determine whether the algorithm can reach the true global optimum, so tests were ran considerably longer than typically needed for solving the problem in question. It is considered to be same if the algorithm converges prematurely in a local optimum closer to or further away from the global optimum. We kept the dimensionality of problems low to prevent running times becoming intolerable.

The population is initialized so that the global optimum is within the initialized range because forcing the algorithm to start outside of the real search range would needlessly complicate the search. It is usually possible to expand the search range to definitely include the optimum and thus avoid wrong initial guesses. The test set also includes problems with a nonsymmetrical optimum to eliminate the possibility of either algorithm benefiting from symmetrical initialization. In addition, a limited test setup was done using initialization that does not include the global optimum to determine how significant the effect of skewed initialization on performance is expected to be.

## 2. Differential Evolution

The DE algorithm [6, 8, 10] was introduced by Storn and Price in 1995 [9]. DE design principles are simplicity, efficiency, and use of floating-point encoding instead of binary numbers.

As in a typical EA, the idea in DE is to have a random initial population that is then improved using selection, mutation, and crossover operations. Control parameters for DE are crossover rate  $CR$ , mutation factor  $F$ , and the population size  $NP$ . A stopping criterion is also needed.

In each generation  $G$ , DE goes through each  $D$  dimensional objective vector  $\vec{x}_{i,G}$  of the population and creates a corresponding trial vector  $\vec{u}_{i,G}$ . The used DE version,

DE/rand/1/bin, is described as follows [8, p. 82]:

$$\begin{aligned}
 & r_1, r_2, r_3 \in \{1, 2, \dots, NP\}, \\
 & \text{(randomly selected, except: } r_1 \neq r_2 \neq r_3 \neq i) \\
 & j_{rand} = \text{int}(\text{rand}_i[0, 1) \cdot D) + 1 \\
 & \text{for}(j = 1; j \leq D; j = j + 1) \\
 & \{ \\
 & \quad \text{if}(\text{rand}_j[0, 1) < CR \vee j = j_{rand}) \quad \cdot (1) \\
 & \quad \quad u_{j,i,G} = x_{j,r_3,G} + F \cdot (x_{j,r_1,G} - x_{j,r_2,G}) \\
 & \quad \text{else} \\
 & \quad \quad u_{j,i,G} = x_{j,i,G} \\
 & \}
 \end{aligned}$$

Indices  $r_1$ ,  $r_2$ , and  $r_3$  are mutually different and drawn from the set of population indices. Parameter  $CR$ , controlling the crossover operation, represents the probability that an element for the trial vector is chosen from a linear combination of three randomly chosen vectors instead of from old objective vector  $\vec{x}_{i,G}$ . Parameter  $F$  is a scaling factor for mutation and its value is typically  $(0, 1+)$ .

The difference between two randomly chosen vectors  $(\vec{x}_{r_1,G} - \vec{x}_{r_2,G})$  defines the magnitude and direction of mutation. When the difference is added to third randomly chosen vector  $\vec{x}_{r_3,G}$ , this corresponds mutation of this third vector. The basic idea of DE is that mutation is self-adaptive to the objective function space and to the current population. At the beginning of generations, the magnitude of mutation is large because vectors in the population are far away in the search space. As evolution proceeds and the population converges, the magnitude of mutation decreases. The self-adaptive mutation of DE enables a global search.

After each mutation and crossover operation, trial vector  $\vec{u}_{i,G}$  is compared to old objective vector  $\vec{x}_{i,G}$ . If the trial vector has equal or lower cost, it replaces the old vector, so the average objective value of the population never increases.

### 3. Modified Generalized Generation Gap Model with Parent-Centric Recombination

One iteration of the modified G3 model with the PCX scheme is described as follows [4]:

1. From the population of individuals, select the best individual and  $\mu - 1$  other individuals randomly to be parents.
2. Generate  $\lambda$  offspring from the chosen  $\mu$  parents using the PCX recombination scheme.
3. Select one individual  $\vec{x}_i$  at random from the population and combine this with  $\lambda$  offspring to make a subpopulation.
4. Select the best solution of the subpopulation and replace the selected individual  $\vec{x}_i$  with this.

In the PCX recombination scheme the offspring  $\vec{y}$  is calculated as follows:

$$\vec{y} = \vec{x}_p + \omega_\zeta \vec{d}_p + \sum_{i=1, i \neq p}^{\mu} \omega_\eta \bar{D} \vec{e}_i. \quad \dots \dots \dots (2)$$

$\vec{x}_p$  is the best individual from the current population (as recommended in [4] for fast convergence),  $\vec{d}_p$  is a direction vector from the mean of  $\mu$  parents to  $\vec{x}_p$ ,  $\bar{D}$  is an average of perpendicular distances to the line defined by  $\vec{d}_p$  from  $\mu - 1$  other parents than  $\vec{x}_p$ , and  $\vec{e}_i$  are  $\mu - 1$  orthonormal bases that span the subspace perpendicular to  $\vec{d}_p$ . Coefficients  $\omega_\zeta$  and  $\omega_\eta$  are zero-mean normally distributed variables with variances  $\sigma_\zeta^2$  and  $\sigma_\eta^2$ .

PCX creates an offspring that has a greater probability of being closer to parent  $\vec{x}_p$  than far from it. Parameters  $\sigma_\zeta^2$  and  $\sigma_\eta^2$  control how much offspring vary from  $\vec{x}_p$  in the direction of  $\vec{d}_p$  and the direction perpendicular to  $\vec{d}_p$ . Because  $\bar{D}$  and length of  $\vec{d}_p$  depend on mutual distances between selected  $\mu$  parents, PCX has the same property of being self-adaptive as DE has.

### 4. DE versus G3 with PCX

In general, DE and G3 with PCX have many similarities that makes their comparison interesting: Both are real-coded EAs with a property of being self-adaptive and both are capable for conducting a rotationally invariant search.

Comparing functional principles of DE (DE/rand/1/bin) and G3 with PCX shows that, G3 is greedier because it always uses the best individual of the population for generating offspring. This increases search speed but also increases the risk of premature convergence.

The formulation of G3 with PCX is more complicated and calculation of one offspring requires more operations than DE does, although overall complexity relative to the size of the population for both methods is  $\Theta(NP)$ . G3 with PCX has two control parameters more to set than DE has.

### 5. Test Arrangements

A set of test runs was conducted with different control parameter settings for both algorithms. With DE, all combinations of 0.2, 0.5, and 0.9 were used for  $F$  and  $CR$  to compare how the algorithm performs with small, medium, and large values for both parameters in different problems, while  $NP$  was varied. Values  $F = 1$  and  $CR = 1$  were not tested because these may lead to stagnation [5].

For the G3 algorithm, the first test set was done by fixing  $\sigma_\zeta = 0.1$  and  $\sigma_\eta = 0.1$ , as suggested by the authors [4]. For  $\lambda$ , values of 2, 4, and 10 and for  $\mu$ , values of 3, 6, and 10 were used. Because the algorithm could not solve some of the problems with any of these combinations, a second test set was conducted, fixing  $\lambda = 2$  and  $\mu = 3$ ,