A Memetic Evolutionary Algorithm

Vu Manh Xuan[1], Nguyen Thanh Thuy[2]

[1] Natural Science Department of Thai Nguyen University
[2] Information Technology Department of Ha Noi University of Technology

Abstract. One of the important properties of evolutionary algorithms is to keep the diversity of the population. This paper presents an algorithm which can be regarded as the integration between Genetic Algorithm (GA) and Evolutionary Strategy (ES). This algorithm has many good properties, especially it satisfies the above important property.

1. Introduction

Evolutionary algorithms can be regarded as the reproduction of the natural evolution process in computer. They also belong to probabilistic algorithms, but the searching solution is found by a population, not by an individual. From an initial population, we can apply genetic operators (selection, crossover, and mutation) to create new individuals which inherit the previous generation's properties and have new properties, then select good individuals for the next generation. Hence, one of the important factors of algorithm is to keep the diversity of the population. In the genetic operators, only crossover and mutation create new individuals. However, algorithms use the different operators as their main evolutionary workhorse, GA specially cares about crossover whereas ES mainly uses mutation.

This paper presents an algorithm that can be regarded as the integration between GA and ES called Blend Evolutionary Algorithm (BEA). Similarly as GA and ES, this algorithm has many good properties, especially it still maintains the diversity of the population. Moreover, BEA has more advantage properties than that of GA and ES.

The paper is organized as follows. In the next section, we review some important aspects of GA and ES. In section 3, we present Blend Evolutionary Algorithm. Section 4 includes experiment results and finally section 5 concludes the paper with a few remarks.

2. Related Works

Binary-coded Genetic Algorithm

Under the initial formulation of GAs, the search space solutions are coded by using the binary alphabet, an individual is binary string of fixed length; genetic operators are selection, crossover and mutation ([2], [3], [6]).

Selection operator includes proportion selection based Roulette wheel, tournament selection and rank selection. Crossover operator includes one-point crossover, N-point crossover and uniform crossover. Mutation operator is bitwise inversion.

Real-Coded Genetic Algorithm (RCGA)

In a RCGA, an individual is represented as an N-dimensional vector of real numbers (x_1, x_2, …, x_N) ([2], [4], [5]). As selection does not involve in the particular coding, no adaptation needs to be made-all selection schemes discussed so far are applicable without any restriction.

Crossover operator for RCGA has many different forms. The following are the two rather popular forms:

Arithmetic crossover ([3]): Given two parents u = $(x_1^1, ..., x_N^1)$; v = $(x_1^2, ..., x_N^2)$, then children are $c_i^1 = \lambda_i x_i^1 + (1 - \lambda_i x_i^2)$ and $c_i^2 = (1 - \lambda_i) x_i^1 + \lambda_i x_i^2$ (for all i=1, ..., N), where λ_i is chosen as a uniformly distributed random value from the interval (0,1).

BLX-α (Blend Crossover) ([1], [5]) crossover creates offspring in the following way: each offspring allele c_i is chosen as a uniformly distributed random value from the interval

$$[\min(x_i^1, x_i^2) - I * \alpha, \max(x_i^1, x_i^2) - I * \alpha],$$

where $I = \max(x_i^1, x_i^2) - \min(x_i^1, x_i^2)$. The parameter α has to be chosen in advance and the best is 0.5.

Mutation operators that are the most common for RCGA are as following ([3], [5]):

Random mutation (uniform): for a randomly chosen gene i of individual x, the allele x_i is replaced by a randomly chosen value from a predefined interval $[a_i, b_i]$.

Non-uniform mutation: assume that T_{max} is the predefined maximum number of generations, the allele x_i is replaced by one of two values

$$x'_i = x_i + \Delta(t, b_i - x_i),$$
$$x''_i = x_i - \Delta(t, x_i - a_i).$$

where random variable $\Delta(t,x)$ determines a mutation step from the range [0, x] in the following way: $\Delta(t, x) = x \cdot \left(1 - \lambda^{(1 - t/T_{max})^r}\right)$. In this formula, λ is a uniformly distributed

random value from the unit interval. The parameter r determines the influence of the generation index t on the distribution of mutation step sizes over the interval [0, x].

Evolutionary Strategies (ESs)

ESs were developed in the late 1960s mainly by Rechenberg independently from Holland's works on genetic algorithms ([5], [6]). Like RCGA, evolutionary strategies aim at solving real-valued optimization problems. In ESs, an individual is represented as a 2N-dimensional real vector which is composed of two vectors (x_1, ..., x_N; σ_1, ..., σ_N). The first half (x_1, ..., x_N) corresponds to the potential solution like in RCGA. The second half (σ_1, ..., σ_N) defines the vector of standard deviations for mutation operation. Unlike GAs, mutation plays a more central role in ESs.

Mutation in ESs is given as:

$$x'_i = x_i + N(0, \sigma_i^2)$$
$$\sigma'_i = \sigma_i.\exp(\tau'.N(0, 1)+\tau.N(0, 1))$$

Selection and sampling in ESs are as the following:

- $(\mu+\lambda)$ES: μ parents are selected from current population, these parents are used to generate a number of λ offsprings. Out of union of parents and offsprings ($\mu+\lambda$ individuals), the best μ are kept for the next generation.
- (μ, λ)ES: in this scheme, μ parents are selected from current population and used to generate λ offsprings ($\lambda \geq \mu$). The parents are discarded completely and the best μ offsprings are kept for the next generation.

3. Blend Evolutionary Algorithm

In this section, we present an algorithm which can be regarded as the integration between Genetic Algorithm (GA) and Evolutionary Strategy (ES) called Blend Evolutionary Algorithm (BEA). Similarly with GA and ES, this algorithm has many good properties, especially it still maintains the diversity of the population. Moreover, BEA has more advantageous properties than that of GA and ES.

The algorithm is as follow:

1. Generate the initial population of M individuals $P=\{x^1, ... , x^M\}$
2. Create population Q of 2M individuals by the following way:
    2.1. Copy M individuals of P to Q;
    2.2. Choose randomly two parents; applying crossover operator to generate one child addition to Q (repeat M times).
3. Each x of Q, creates an offspring x' by mutation operator, choose best one of {x, x'} replaced x.
4. Select the M individuals out of Q to replace P by tournament selection.
5. Stop if stopping criterion is satisfied; otherwise, go to step 2.

This algorithm uses both crossover and mutation in two phases. First crossover operator performs M times, two parents create only one child, which is added in intermediate population. Then mutation operator performs with all individuals of population by schema (1+1)ES. The crossover is arithmetic crossover; the mutation is uniform mutation. However, can used the another forms of crossover or mutation.

Therefore, the individuals of intermediary population do not only inherit their parents' properties, but also change with high random which makes the higher diversity by the mutation operator.

So BEA is not only similar to RCGA in coding and the way of using crossover, but is like as ES in using mutation and generative scheme. Moreover, BEA is different from RCGA, it does not set up a big size population but still maintains the population's diversity. It is also different from ES in performing crossover operator for the setting up it's intermediary population step to inherit the parents' good characteristics of the former population.

4. Experiment

Test Functions

Test functions included: $f_1(x)$ (Sphere function); $f_2(x)$; $f_3(x)$ (Rosenbrock function); $f_4(x)$ (Rastringin function); $f_5(x)$ (Griewangk function) and $f_6(x)$ function ([1], [3], [6]). Table 1 shows that functions.

Table 1. Test functions

Function	n	S	f_{min}
$f_1(x) = \sum_{i=1}^{n} x_i^2$	30	[-5.12, 5.12]	0
$f_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	[-5.12, 5.12]	0
$f_3(x) = \sum_{i=1}^{n-1} (100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$	30	[-5.12, 5.12]	0
$f_4(x) = 10n + \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i))$	30	[-5.12, 5.12]	0
$f_5(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}}) + 1$	30	[-5.12, 5.12]	0
$f_6(x) = \sum_{i=1}^{n} (-x_i \sin\sqrt{\lvert x_i \rvert})$	30	[-500,500]	-12569.5

- f_1 is a continuous, strictly convex, and uni-modal function.
- f_2 is a continuous, and unimodal function with one global minimum.
- f_3 continuous, and unimodal function. Its difficulty concerns the fact that searching along the coordinate axes only gives a poor rate of convergence, since the gradient is not oriented along the axes.
- f_4 is a scalable, continuous, and multimodal function, which is made from f_1 by modulating it with $10\cos(2\pi x_i)$.
- f_5 is a continuous, and multimodal function. This function is difficult to optimize because it is non-separable and search algorithm has to climb a hill to reach the next valley.
- f_6 is multimodal function and the global minimum is at (420.968, ..., 420.968), very close to one corner of the search space.

Experiment Setup

Because the BEA is regarded as the RCGA's improvement, the experiment program is compared with RCGA. The parameters include:

RCGA with the fixed population size of 20 and 100 individuals. Each individual is n-dimensional real vector. The program repeated 30 runs independently uses crossover with probability $p_c = 0.9$, and the uniform mutation operator with probability $p_m = 0.1$. In each runs, first created randomly initial population, then performed 2500 times, in each step chosen random two parents, used crossover operator generate two offsprings with probability p_c, mutation with probability p_m one of these offsprings and chosen two best individuals from parents and offsprings replaced two parents. In the first experiments, we used arithmetic crossover and uniform mutation. In the second experiments, we used BLX-α crossover and uniform mutation.

BEA has the population size of 10 individuals (M=10) which use the above genetic operators. It performs 30 runs independently, each runs 100 generations. In each generation, chosen two parent randomly, used crossover operator generate one offspring, this offspring adds to intermediary population; then each individual of intermediary population performs mutation generate one offspring, if new individual is better then replaced to old individual. Then, M individuals from intermediary population by the way tournament selection are kept for the next generation.

The results are shown in table 2 and table 3. In this tables, columns (2), (5) and (8) show the average time of each run. Columns (3), (6) and (9) show average value of best individual of each run. Columns (4), (7) and (10) show the function value of best individual of 30 runs.

Results

Table 2 shows the experiment's results, in this experiment used arithmetical crossover and uniform mutation. RCGA with population's size M=20 and M=100 individuals.

Table 2. Result of experiment used arithmetical crossover.

	BEA (M=10)			RCGA (M=20)			RCGA (M=100)		
	Time	Average	Best	Time	Average	Best	Time	Average	Best
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
$f_1(x)$	2.80	0.075	0.017	4.41	5.981	2.784	4.08	1.8529	1.1945
$f_2(x)$	3.02	0.829	0.568	4.85	8.873	5.510	4.68	5.2765	3.5479
$f_3(x)$	2.83	106.27	35.951	4.30	1173.153	506.290	4.16	266.392	156.34
$f_4(x)$	3.13	14.697	7.353	5.00	80.722	44.246	4.95	106.3285	58.007
$f_5(x)$	3.28	0.011	0.002	5.04	0.263	0.141	4.93	0.09925	0.0461
$f_6(x)$	3.68	-12083	-12429	5.84	-5039.29	-6641.2	5.76	-3338.7	-4277.8

Table 3 shows the experiment's results, in this experiment used BLX-α crossover (α=0.5) and uniform mutation. RCGA with population's size M=20 and M=100 individuals.

Table 3. Result of experiment used BLX-α crossover (α=0.5).

	BEA (M=10)			RCGA (M=20)			RCGA (M=100)		
	Time	Average	Best	Time	Average	Best	Time	Average	Best
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
$f_1(x)$	2.96	0.0466	0.0136	4.68	4.4899	1.5331	4.30	0.8599	0.3843
$f_2(x)$	3.28	0.5457	0.2872	4.88	4.7829	2.7015	4.85	2.7552	2.0459
$f_3(x)$	2.94	146.401	61.538	4.55	939.1193	267.55	4.36	224.7713	103.02
$f_4(x)$	3.39	15.7433	6.7345	5.15	74.0886	46.952	5.36	181.3437	152.19
$f_5(x)$	3.78	0.0259	0.0009	5.35	0.2042	0.0583	5.21	0.0484	0.0230
$f_6(x)$	3.82	-12425.6	-12513	5.77	-8692.7	-10655	6.12	-4698.4	-6086.1

This result shows that with the shorter time, BEA gives better results than that of RCGA and used BLX-α crossover better than arithmetical crossover.

5. Discussion of Results

Since the searching solution of evolutionary algorithms is searched by a population, the diversity of the population is an important factor. GAs specially care about crossover whereas ES mainly uses mutation, whereas BEA uses both operators in two phases with probability 1. RCGA with large population's size (higher diversity) has better result, whereas BEA needs little population's size. The tournament selection operator increases the diversity of the population and it also increases the speed of population convergence.

This idea can be further advanced by choosing genetic operators or evolutionary strategies, for example $(\mu+\lambda)$ES with $\mu, \lambda > 1$. This algorithm can also use another forms of crossover, mutation operator or selection. We preserve these as our future work.

Acknowledgement

The authors would like to thank Dr Nguyen Xuan Hoai for his useful suggestions and discussions during the completion of this paper.

References

[1] Kalyanmoy Deb, Hans-Georg Beyer (1999), *Self-Adaptive Genetic Algorithms with Simulated Binary Crossover*, Technical Report No. CI-61-99.
[2] Goldberg, D.E. (1989), *Genetic algorithms in search, optimization and machine learning*, Addison-Wesley, Reading, MA.
[3] Michalewicz, Z. (1992), *Genetic Algorithms + Data Structures = Evolution Program*. Springer Verlag.
[4] M. Lozano,F. Herrera, Natalio Krasnogor, Daniel Molina (2004), *Real-Code Memetic Algorithms with Crossover Hill-Climbing*, Evolutionary Computation 12(3), 273-302.
[5] Unlrich Bodenhofer (2004), *Genetic Algorithms: Theory and Applications*, Lecture Notes.
[6] Xin Yao and Yong Liu (1997), *Fast Evolution Strategies*, Contr. Cybern. Vol26, 467-496.