An Improved Genetic Algorithm for Nonlinear Programming Problems

Weiyi Qian
Department of Mathematics
Bohai University
Jinzhou, 121000 China
qianweiyi2008@163.com

Guojuan Chu
Department of Mathematics
Bohai University
Jinzhou, 121000 China
chuguojuan12@163.com

Abstract

In this paper, an improved genetic algorithm is proposed for nonlinear programming problems. In this algorithm, each individual is taken as a particle with mass. The mass of each individual is defined and the center of gravity is computed according to physical formula. A simplex is formed randomly from the population. One of the points of the simplex is reflected in the center of gravity of the remaining points to obtain a new trial point. The crossover operator, based on above method, is modified to improve the efficiency of genetic algorithm. To evaluate the efficiency of the improved algorithm, the algorithm is applied to five test problems, and our results are compared with other methods. The numerical results illustrate the efficiency of our method.

1. Introduction

Here we consider the global nonlinear programming problems in the following form

$$\min f(x) \quad s.t.\ x \in D \tag{1}$$

where $x \in R^n$, $f : R^n \to R$ is nonlinear function, and $D = \{x \in R^n \mid a^i \le x^i \le b^i, a^i, b^i \in R, i = 1, 2, \cdots, n\}$. Such problem arises in many practical applications, e.g. in the risk management, economics, data analysis, engineering design, financial planning, etc. Most cases of practical interest are characterized by multiple local optima. However, most optimization techniques currently used are local methods, which easily fail, and at best, find only one local optimum. The objective of this study is to develop a global optimization algorithm for solving the problem (1).

A global optimization algorithm aims at finding a global minimizer, or its close approximation of the objective function f. A point x^* is said to be a global minimizer of f if $f^*=f(x^*)\leqslant f(x)$, $\forall x \in D$. For practical interests, especially in applied sciences and in engineering, it is required that an approximation $\hat{x}^*$ of a global minimizer of f be found with $|f^* - f(\hat{x}^*)| < \varepsilon$. However, in many applications the function of interest is not differentiable. In recent year, different stochastic optimization methods are proposed for solving global optimization problems. These approaches can roughly be classified into following categories: random samp-ling, evolutionary algorithms, simulated annealing, particle swarm algorithms, and so on. In our study, we focus on one of these methods, namely genetic algorithm (GA).

The GA, originally developed by Holland[1], proved to solve various combinatorial optimization problems efficiently. Later, some authors have proposed GAs for continuous variables[2][3][4][5] . In recent years, in order to improved effectiveness of the GA , many heuristic methods have been incorporated with the GA's population initialization, reproduction and selection, fitness evaluation, crossover or mutation and so on. Recently, Hu, et al has proposed a new version of real coded GA[6], we call it GA1. But GA1 has three drawbacks. The first drawback is that the selection criterion is too biased towards the better points and therefore it is too greedy and may only work for easy problems[7]. The second drawback is that the mutation operator is rather randomly chosen. If the difference between b^i and a^i is high then this will create a point far away from the point generated by crossover operation[7]. The third is that the crossover operator can not make GA more exploratory[7]. Ali et al has been proposed modifications to GA1[7] , we call it GA2. In GA2, the main modification is that one of the points of the simplex is reflected in the centroid of the remaining point to obtain the new trial point in crossover.

In this paper, we take each point as a particle with mass and define the mass of each point. The center of gravity is computed according to physical formula. For

978-0-7695-3736-8/09 $25.00 © 2009 IEEE
DOI 10.1109/ICNC.2009.281

problem (1), an improved genetic algorithm is proposed, we call it GA3. The main difference between GA3 and GA2 is that the center of gravity replaces the centroid. The GA3 is applied to five test problems, and our results are compared with GA1 and GA2. The numerical results illustrate the efficiency of our method.

2. Center of gravity calculation

Assume the population has N points, $x_1, x_2, \cdots, x_N$, we can think of each point as a particle with mass. The mass of each point x_i, m_i, is evaluated by using the objective function value of the point, x_i, relative to the objective function value of the current best point.

$$m_i = \exp\left[-n \times \frac{f(x_i) - f(x_{best})}{\sum_{k=1}^{N} \left[f(x_k) - f(x_{best}) \right]} \right], i = 1,2,\cdots,N \quad (2)$$

where $x_{best} = \arg\min\{f(x_i) \mid i = 1,2,\cdots,N\}$ is the current best point. From (2), we can see that the smaller the objective function value of the point, x_i, is, the bigger the mass is. Reversely, the bigger the its objective function value is, the smaller the mass is.

After determining the mass of each point, the center of gravity G of n points, $x_1, x_2, \cdots, x_n$, of the population , is computed as follows

$$G = \frac{\sum_{j=1}^{n} m_j x_j}{\sum_{j=1}^{n} m_j} \quad (3)$$

3. Improved genetic algorithm(GA3)

The GA consist of the binary coded GA and real coded GA. The real coded GA is superior to its conventional binary coded counterpart[6][8]. Several versions of real coded GA have been proposed [6][9][10]. Recently, Hu et al have proposed a new version of real coded GA. Numerical studies were carried out on a relatively large set of test problems with the conclusion that the convergence of the algorithm is rapid. The real coded genetic(GA1) proposed by Hu can be described as follows .

The Genetic Algorithm(GA1)

Step 1 Generate the initial population P

$$P = \{x_1, x_2, \cdots, x_N\}$$

were the individual x_i , $i = 1,2,\cdots,N$ are sampled randomly in D, evaluate $f(x)$, at each x_i , i=1,2, ⋯, N. Take $N >> n$. Set generation counter $k = 0$.

Step 2 Determine best, bad individual in P. Determine the individuals x_{best}, x_{bad} and their function values f_{best}, f_{bad} such that

$$f_{best} = \min_{x \in P} f(x) \text{ and } f_{bad} = \max_{x \in P} f(x)$$

If the stopping condition(e.g. $f_{bad} - f_{best} < \varepsilon$) is , then stop.

Step 3 Generate new individuals to replace individuals in P.

(1) Selection: using tournament selection to select $m < N$ from P as parents.

(2) Crossover: take two selected individuals

$$x_i = (x_i^1, x_i^2, \cdots, x_i^N)$$

and

$$x_j = (x_j^1, x_j^2, \cdots, x_j^N)$$

from parents and generate two new individuals as follows

$$\begin{aligned} \tilde{x}_i^l &= \alpha_l x_i^l + (1 - \alpha_l) x_j^l \\ \tilde{x}_j^l &= \alpha_l x_j^l + (1 - \alpha_l) x_i^l \end{aligned}, \; l = 1,2,\cdots,n \quad (4)$$

where α_l are uniform random numbers in [-0.5, 0.5]. The process is repeated until m new feasible individuals are generated.

(3) Mutation: for each child $\tilde{x}_j$ generated, the mutation is carried out with probability P_u by setting

$$\tilde{x}_j^i = \tilde{x}_j^i + \gamma \times (b^i - a^i) \quad (5)$$

for a randomly chosen component $\tilde{x}_j^i$ of $\tilde{x}_j$, where $\gamma = 0.1$, b^i and a^i are the upper and lower bound of the element x^i .

Step 4 Update P: the m children replace the m worst individuals. Set k=k+1, and go to step 2.

Ali and TÖrn proposed a modified genetic algorithm(GA2) to GA1[7]. We now describe their works. They replaced the tournament selection with random selection, and modified the mutation operator(5) to

$$x^i = x^i + \gamma(b^i - a^i) \quad (6)$$

where γ is a random number in [−0.01, 0.01]. The crossover operator was modified too, its method can be described as follows. GA2 creates two children from randomly selected n+2 parents, x_{p1}, x_{p2}, ⋯, $x_{p(n+1)}$, $x_{p(n+2)}$, form P. The selected n+2 points are used to calculate the centroid G of the n points remaining after excluding the two worst points, say $x_{p(n+1)}$ and $x_{p(n+2)}$. The centroid G is given by

$$G = \frac{1}{n}\sum_{j=1}^{n} x_{p(j)} \tag{7}$$

The first child is taken as the best point of $\{\hat{x}_1, \hat{x}_2\}$, where

$$\hat{x}_1 = 2\times G - x_{p(n+1)} \text{ and } \hat{x}_2 = 2\times G - x_{p(n+2)} \tag{8}$$

If the j-th point (j=1,2) $\hat{x}_j$ is not in D then it is calculated as

$$\hat{x}_j = \frac{1}{2}(G + x_{p(n+j)}) . \tag{9}$$

The second child is found from the best point of $\{\hat{x}_3, \hat{x}_4\}$, where $\hat{x}_3$ and $\hat{x}_4$ are obtained using the crossover rule (4). This crossover is carried out between two parents selected randomly from the n+2 points, again excluding two worst points, say $x_{p(n+1)}$ and $x_{p(n+2)}$. If the trial points fall outside D, random selection of $\alpha_i \in [-0.5, 0.5]$ continues until $\hat{x}_3$, $\hat{x}_4 \in D$. For the next pair of children n+2 points are again selected randomly from P and above process continued.

The center in GA2 is the center of geometrical, it has nothing to do with the objective function value. The center of gravity G given by (3) is closely related to the objective function value and it is biased towards the better points of n points. Hence, in order to make the GA2 more efficient, an improved genetic algorithm (GA3) to GA2 is proposed. The GA3 differs from GA2 in two aspects. First, we make a random selection of n+1 points in P and include the current best point bringing the number of points up to n+2. Therefore, the modification is that the n+1 distinct parents, x_{p2}, $x_{p3}, \cdots, x_{p(n+1)}, x_{p(n+2)}$, are sampled from P and let x_{p1} denote the best point. Assume $x_{p(n+1)}$ and $x_{p(n+2)}$ are two worst points, we calculate the mass of $x_{p(i)}$ (i=1,2, $\cdots$, n) by (2) and the center of gravity by (3). Second, the crossover rule (8) is replaced in the following form

$$\hat{x}_1 = \begin{cases} 2\times G - x_{p(n+1)}, & \text{if } f(G) \le f(x_{p(n+1)}) \\ 2\times x_{p(n+1)} - G, & \text{if } f(x_{p(n+1)}) < f(G) \end{cases} \tag{10}$$

and

$$\hat{x}_2 = \begin{cases} 2\times G - x_{p(n+2)}, & \text{if } f(G) \le f(x_{p(n+2)}) \\ 2\times x_{p(n+2)} - G, & \text{if } f(x_{p(n+2)}) < f(G) \end{cases} . \tag{11}$$

The rule (10) and (11) are introduced in the crossover phase to make GA3 more exploratory and the new points bias to the better points. The others, such as the section、mutation and the second child and so on, are the same as GA2.

4. Numerical results

To evaluate the efficiency of the GA3. we select five test problems[11] , see table 1. In table 1, n denotes the dimension of the problem, D the domain of each variable, NLM number of known local minima, f^* the global minimum.

Table 1. The test problems

Problems	n	D	NLM	f^*
Shekel5	4	$0 \le x_i \le 10$	4	-10.1532
Shekel7	4	$0 \le x_i \le 10$	7	-10.4029
Shekel10	4	$0 \le x_i \le 10$	10	-10.5364
Hartman3	3	$0 \le x_i \le 1$	4	-3.8627
Hartman6	6	$0 \le x_i \le 1$	4	-3.3223

Table 2. Results of the GA1

	FE	cpu	TS
Shekel5	1374	0.19	15
Shekel7	1463	0.24	21
Shekel10	1702	0.31	26
Hartman3	578	0.14	92
Hartman6	1814	0.58	100

Table 3. Results of the GA2

	FE	cpu	TS
Shekel5	2163	0.34	67
Shekel7	2731	0.49	80
Shekel10	3384	0.61	85
Hartman3	1136	0.29	99
Hartman6	3597	1.18	100

Table 4. Results of the GA3

	FE	cpu	TS
Shekel5	1864	0.28	66
Shekel7	2702	0.42	82
Shekel10	2986	0.54	83
Hartman3	953	0.21	100
Hartman6	2897	0.87	100

We compare GA1, GA2 and GA3 using above problems for 50 independent test runs. The results are shown in table 2 , table 3 and table 4. In those tables, FE denotes the average number of function evaluations, cpu the average runtimes, TS the percentage of success. Some parameters are given as follows: the population size N=12n, mutation probability P_u=0.001, the number of children m is the nearest even integer to 0.1N. Table 2 and table 3 show that number of successes of GA2 superior to that of GA1. But FE and cpu needed by GA2 are nearly twice that by GA1. From table 2, table 3 and table 4, we can see that number of successes of GA3 superior to that of GA1 and is nearly the same as that of GA2. FE and cpu needed by GA3 superior to that by GA2, but inferior to that by GA1. The numerical results illustrate the efficiency of our method.

5. Conclusions

In this paper, we have shown that the improved genetic algorithm can be efficiently applied to the global optimization of the nonlinear programming problems with box constraints. Our main contributions defines the center of gravity and use new crossover operator based on the center of gravity in improved genetic algorithm. The improved genetic algorithm is compared with GA1 and GA2. The numerical results illustrate that our method superior to GA1 in number of successes of algorithm, and it superior to GA2 in FE and cpu needed by algorithm. Consequently, we feel that the method could be used as a general purpose global optimization technique. Research is continuing to develop even more efficient GA.

Acknowledges

This research was supported by the National Science Foundation of China (10871033), the National Science Foundation of China (10671126), and Educational Department Foundation of Liaoning province (20080004).

References

[1] J.H. Holland, "Outline for a logical theory of adaptive systems", *Journal of the Association for Computing Machinery*, 1962, 93(3), pp.297-314.

[2] J. Tang, and D. Wang, "A hybrid genetic algorithm for a type of nonlinear programming problems", *Computers and Mathematics with Application,* 1998, 36(5), pp.11-21.

[3] R. Chelouah and P. Siarry, "continuous genetic algorithm designed for the global optimization of multimode function", *Journal of Heuristicsl*, 2000, 6, pp.191-213.

[4] J. J. Mendes, J. F. Goncalves and M. G. C. Resendc, "A random key based genetic algorithm for the resource constrained project scheduling problem", *Computer and Operations Research*, 2009, 36, pp. 92-109.

[5] W. Y. Liang and C. C. Huang, "A hybrid approach to constrained evolutionary computing case of product synthesis", *Omega*, 2008, 36, pp.1072-1085.

[6] Y.F. Hu, K. C. Maguire, D. Cokljat, and R. J. Blak,. "Parallel controlled random search algorithms for shape optimization", *Proceeding of the Parallel CFD Conferences, Manchester United Kingdom,* May,1997, pp. 345-352.

[7] M.M. Ali, and A. TÖrn, "Population set based global optimization algorithms: some modifications and numerical studies", *Computers and Operations Research*, 2004,31(10), pp. 1703-1725.

[8] Z. Michalewicz. "Genetic algorithms, numerical optimization, and constraints", *In: Proceedings of the sixth International Conference on Genetic Algorithms, San Mateo, Morgan Kaufmann Publishers*, April, 1995, 151-158.

[9] J. Yen and B. Lee, "A Simplex Genetic Algorithm Hybrid", *Proceedings of the 1997 IEEE International Conference on Evolutionary Computing,* Indianapolis, Indiana, April, 1997, pp. 175-180.

[10] R. Yang and I. Douglas, "Simple Genetic Algorithm with Local Tuning : Efficient Global Optimizing Technique", *Journal of Optimization Theory and Applications*, 1998, 98(2), pp. 449-465.

[11] L. Wang, *Interlligent Optimization Algorithms with Applications*, Tsinghua university Press, Beijing, 2003, pp.5-20.