An Evolutionary Algorithm Based on Stochastic Weighted Learning for Constrained Optimization

Jun Ye[1], Xiande Liu[1], and Lu Han[2]

[1] Optics Engineering Dept.,
Huazhong Univ. of Science & Tech.,
430074 Hubei, P.R. China
yejun_clinux@hotmail.com
[2] Electronics & Info. Engineering Dept.,
Huazhong Univ. of Science & Tech.,
430074 Hubei, P.R. China

Abstract. In this paper, we propose an evolutionary algorithm based on a single operator called stochastic weighted learning, i.e., each individual will learn from other individuals specified with stochastic weight coefficients in each generation, for constrained optimization. For handling equality and inequality constraints, the proposed algorithm introduces a learning rate adapting technique combined with a fitness comparison schema. Experiment results on a set of benchmark problems show the efficiency of the algorithm.

1 Introduction

Most engineering optimization problems include equality and/or inequality constraints, and recent years, evolutionary algorithms have received a lot of attention regarding their potential for solving effectively such constrained optimization problems (see [1], [2], [3] for a comprehensive survey).

In a previous work [4], we have introduced a new evolutionary algorithm based on a single operator called stochastic weighted learning for unconstrained optimization problems. The idea of the algorithm is very simple, i.e., in each generation each individual will learn from other individuals in the population specified with stochastic weight coefficients that represent the learning strength related to them. The similar strategy learning process can be commonly observed in the behavior of rational agents within economic environment, and the operator tries to mimic such process.

In this paper, we attempt to extend our algorithm to solve constrained optimization problems. Section 2 presents the basic structure of the proposed algorithm. Section 3 introduces a learning rate adapting technique combined with a fitness comparison schema for handling equality and inequality constraints. Section 4 presents the experimental results on a set of benchmark problems. Comparisons with other evolutionary algorithms are also included in this section. Finally, Section 5 concludes with a brief summary of the paper.

L. Wang, K. Chen, and Y.S. Ong (Eds.): ICNC 2005, LNCS 3611, pp. 1105 – 1111, 2005.

2 New Evolutionary Algorithm Based on Stochastic Weighted Learning

The general nonlinear programming problem can be formulated as follows:

$$\max f(x)$$
$$s.t.\begin{cases} g_j(x) \le 0 & j = 1,\ldots, p \\ h_j(x) = 0 & j = p+1,\ldots, q \end{cases} \tag{1}$$

where $x = (x_1, x_2, \ldots, x_n) \in R^n$, $x_i \in [x_i^l, x_i^u]$, $i = 1, 2, \ldots, n$, is n-dimensional real vector, $f(x)$ is the objective function, $g_j(x)$ is the jth inequality constraint, $h_j(x)$ is the jth equality constraint, and $D = \prod[x_i^l, x_i^u] \subseteq R^n$ defines the search space.

Unlike most EAs that have different selection strategies, mutation rules and crossover operators, the proposed algorithm uses only one operator that mimics the strategy learning process of rational agents to achieve the objective of optimization, therefore it is fairly simple and can be easily realized.

2.1 Individual Representation

Individual representation in the proposed algorithm is straightforward. Each individual in the population is represented only by its solution variables. We denote the ith individual in the tth generation by $x(t)_i$, where $i = 1, 2, \ldots, M$, M is the population size.

2.2 Stochastic Weighted Learning

In each generation, each individual will learn new strategy profile from other m individuals for next generation. In these m individuals, $(m - 1)$ individuals are randomly selected from the whole population and one is the best-fit individual in this generation. Each one of these m individuals is specified with a weight coefficient that represents the positive or negative learning strength related to that individual. We denote these weight coefficients by w_k, where $k = 1, 2, \ldots, m$, and they are satisfied the relation below:

$$\sum_{k=1}^{m} w_k = 1 \tag{2}$$

where $w_k \sim U(-1, 1)$. The strategy profile learned from these m individuals is then defined as follows:

$$x_i' = \sum_{k=1}^{m} w_k x(t)_k \tag{3}$$

where $i = 1, 2, \ldots, M$.

If a certain component of the strategy profile learned, say x'_{ij} (the jth component of x'_i), is outside the parametric bounds defined by the problem, the algorithm will use the arithmetical average of the corresponding components of these m individuals instead.

Each individual in the population will adopt the new strategy profile in next generation if the fitness of the new strategy profile is greater than its current one. Otherwise, it will hold the current strategy profile without any change.

2.3 Algorithm

We have already explained each element in our algorithm. The pseudocode of the proposed algorithm can be summarized as follows.

```
Procedure SWL
  t = 0;
  initialize x(0);
  while t < T do
    find the best-fit individual;
    for i = 1, 2, ..., M do
      randomly select (m-1) individuals and generate w_k;
      x•_i = sum[w_k x(t+1)_k], k = 1, 2, ..., m;
    end
    for i = 1, 2, ..., M do
      if fitness[x•_i] > fitness[x(t)_i] then
        x(t+1)_i = x•_i;
      else
        x(t+1)_i = x(t)_i;
      end
    end
    t = t + 1;
  end
end
```

3 Constraint Handling

Here we introduce a learning rate adapting technique combined with a fitness comparison schema for handling equality and inequality constraints.

3.1 Fitness Comparison

All equality constraints are converted into inequality constraints, $|h(x)| - \varepsilon \le 0$, where ε is the degree of tolerated violation. We define $v(x) = \max(v_1(x), \ldots, v_q(x))$, where $v_j(x)$ is:

$$v_j(x) = \begin{cases} \max(g_j(x),0) & j = 1,\ldots,p \\ \max(|h_j(x)| - \varepsilon,0) & j = p+1,\ldots,q \end{cases} \tag{4}$$

In the algorithm, the fitness of x_1 is greater than the fitness of x_2 if one of the conditions (a) $v(x_1) = 0$ and $v(x_2) = 0$ and $f(x_1) > f(x_2)$; or (b) $v(x_1) = 0$ and $v(x_2) > 0$; or (c) $v(x_1) > 0$ and $v(x_1) < v(x_2)$ is fulfilled. That is, if the both individuals are feasible, then the one with greater $f(x)$ is better; feasible individual is better than infeasible one; and if the both individuals are infeasible, then the one with less $v(x)$ is better.

3.2 Learning Rate Adapting

In every Δt generations, there are $\Delta t \times M$ times of tries for learning a better strategy profile. We denote the times of made-learning (the strategy profile learned is better than the current one) in these tries by s. The learning rate r is defined as:

$$r = s / (\Delta t \times M) \tag{5}$$

In the algorithm, the number of individuals that each individual will learn from, m will be adjusted adaptively within a predefined range $[m_1, m_2]$ along the evolution according to r, i.e., if $r < r_1$ and $m > m_1$, then $m = m - 1$; else if $r > r_2$ and $m < m_2$, then $m = m + 1$. (r_1 and r_2 are two predefined threshold values, $0 < r_1 < r_2 < 1$)

For the problems that have equality constraints, the degree of tolerated violation ε will be adjusted adaptively along the evolution according to r as well, i.e., if $r < r_1$, then $\varepsilon = \varepsilon / \alpha$; else if $r > r_2$, then $\varepsilon = \varepsilon \times \alpha$. ($\alpha$ is a predefined scaling factor, $0 < \alpha < 1$)

The idea is, start with an initial value $m^{(0)}$ (and $\varepsilon^{(0)}$ if have equality constraints), for every Δt generations, if the learning rate is lower than a certain level, then decrease m to speed up the learning process (and increase ε to enlarge the feasible domain); or if the learning rate is higher than a certain level, then increase m to slow down the learning process (and decrease ε to reduce the feasible domain).

4 Experimental Results

We use a set of benchmark problems G01 to G13 for testing the performance of the proposed algorithm. These problems are proposed in [1] and [5]. And for all test problems, the parameters are fixed to $M = 70$, $T = 5000$, $\Delta t = 10$, $r_1 = 0.01$, $r_2 = 0.03$, $m^{(0)} = 10$, $m_1 = 9$, $m_2 = 11$, $\varepsilon^{(0)} = 10$, $\alpha = 0.85$. These parameters were selected based on the experimental experience. A total number of 35 independent runs are executed for each problem, and each run involves 350000 function evaluations. Results are shown in Table 1.

The column indicated by "optimal" in Table 1 shows the known "optimal" solution for each problem. The three columns under caption "Fitness" give the best, mean, and worst objective value found, and the three columns under caption "Violation" give the minimal, mean, and maximal solution violations.

From the table, one may find that the algorithm has consistently found the "optimal" solutions for all problems in 35 runs except G02. And for problems G03, G05, G11 and G13 that have equality constraints, the solution violations are very low, compared with 1e-3 and 1e-4, the values of tolerated violation degree usually used by other evolutionary algorithms.

Table 2 shows the result of another experiment on problem G02 with the parameter settings of $M = 200$, $T = 12000$, $m^{(0)} = m_1 = m_2 = 11$. A total number of 35 independent runs are executed, and each run involves 2400000 function evaluations. As expected, with large population and suitable value m, the proposed algorithm can consistently found the optimal solutions for G02 in all runs.

Table 3 summarizes the comparison between our results and Runarsson and Xin Yao's results [6]. The results of [6] shown in Table 3 is the statistics of 30 independent

Table 1. Experimental Results on Benchmark Problems

		Fitness			Violation		
	optimal	Best	Mean	Worst	Best	Mean	Worst
G01	-15.000	-15.000	-15.000	-15.000	0	0	0
G02	-0.803619	-0.803619	-0.792695	-0.756970	0	0	0
G03	-1.000	-1.000	-1.000	-1.000	0	1.33e-17	2.22e-16
G04	-30665.539	-30665.539	-30665.539	-30665.539	0	0	0
G05	5126.498	5126.498	5126.498	5126.498	0	2.73e-13	4.55e-13
G06	-6961.814	-6961.814	-6961.814	-6961.814	0	0	0
G07	24.306	24.306	24.306	24.306	0	0	0
G08	-0.095825	-0.095825	-0.095825	-0.095825	0	0	0
G09	680.630	680.630	680.630	680.630	0	0	0
G10	7049.248	7049.248	7049.248	7049.248	0	0	0
G11	0.750	0.750	0.750	0.750	0	7.22e-17	1.11e-16
G12	-1.000000	-1.000000	-1.000000	-1.000000	0	0	0
G13	0.053950	0.053950	0.053950	0.053950	0	1.53e-15	4.88e-15

Table 2. Experimental Result on G02 with $M = 200$, $T = 12000$, $m^{(0)} = m_1 = m_2 = 11$

		Fitness			Violation		
	optimal	Best	Mean	Worst	Best	Mean	Worst
G02	-0.803619	-0.803619	-0.803619	-0.803619	0	0	0

Table 3. Comparison Between Our (Indicated by SWL) and Runarsson and Xin Yao's (Indicated by SR [6]) Algorithms

		Best		Mean		Worst	
	optimal	SWL	SR	SWL	SR	SWL	SR
G01	-15.000	-15.000	-15.000	-15.000	-15.000	-15.000	-15.000
G02	-0.803619	-0.803619	-0.803515	-0.792695	-0.781975	-0.756970	-0.726288
G03	-1.000	-1.000	-1.000	-1.000	-1.000	-1.000	-1.000
G04	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539
G05	5126.498	5126.498	5126.497	5126.498	5128.881	5126.498	5142.472
G06	-6961.814	-6961.814	-6961.814	-6961.814	-6875.940	-6961.814	-6350.262
G07	24.306	24.306	24.307	24.306	24.374	24.306	24.642
G08	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825
G09	680.630	680.630	680.630	680.630	680.656	680.630	680.763
G10	7049.248	7049.248	7049.316	7049.248	7559.192	7049.248	8835.655
G11	0.750	0.750	0.750	0.750	0.750	0.750	0.750
G12	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000
G13	0.053950	0.053950	0.053957	0.053950	0.067543	0.053950	0.216915

runs and each run involves 350000 function evaluations. And for problems G03, G05, G11 and G13, the tolerated violation degree for equality constraints is fixed to 10^{-4}. From the table, one may find that the algorithm proposed in this paper outperforms that in [6] for all cases (the best solution found by [6] for G05 which is better than the "optimal" solution is a result of loose tolerated violation degree).

Table 4 and 5 summarizes the comparisons between our results and the more recent work [7]. The results of [7] shown in Table 4 and 5 is the statistics of 31 independent runs and each run involves 1500000 function evaluations. From the tables, one may find that the results of our algorithm are better not only in term of objective value but also in term of solution violations.

Table 4. Comparison Between Our (Indicated by SWL) and Hamida and Schoenauer's (Indicated by ASCHEA [7]) Algorithms

		Best		Median		Mean	
	optimal	SWL	ASCHEA	SWL	ASCHEA	SWL	ASCHEA
G01	-15.000	-15.000	-15	-15.000	-15	-15.000	-14.84
G02	-0.803619	-0.803619	-0.803614	-0.794885	-0.794568	-0.792695	-0.788950
G03	-1.000	-1.000	-1.000	-1.000	-0.999995	-1.000	-0.99997
G04	-30665.539	-30665.539	-30665.5	-30665.539	-30665.5	-30665.539	-30665.5
G05	5126.498	5126.498	5126.5	5126.498	5126.5	5126.498	5126.53
G06	-6961.814	-6961.814	-6961.81	-6961.814	-6961.81	-6961.814	-6961.81
G07	24.306	24.306	24.3323	24.306	24.6162	24.306	24.6636
G08	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825
G09	680.630	680.630	680.630	680.630	680.635	680.630	680.641
G10	7049.248	7049.248	7049.42	7049.248	7272.19	7049.248	7615.2
G11	0.750	0.750	0.75	0.750	0.75	0.750	0.75
G12	-1.000000	-1.000000	-	-1.000000	-	-1.000000	-
G13	0.053950	0.053950	-	0.053950	-	0.053950	-

Table 5. Violation Comparison Between Our (Indicated by SWL) and Hamida and Schoenauer's (Indicated by ASCHEA [7]) Algorithms for G03, G05, G11 and G13

	Violation					
	Best		Mean		Worst	
	SWL	ASCHEA	SWL	ASCHEA	SWL	ASCHEA
G03	0	0	1.33e-17	1.8e-16	2.22e-16	2.22e-16
G05	0	1.6e-9	2.73e-13	2.61e-4	4.55e-13	0.008
G11	0	0	7.22e-17	1.87e-11	1.11e-16	4.46e-11
G13	0	-	1.53e-15	-	4.88e-15	-

5 Conclusion

This paper is a continuation of the study devoted to stochastic weighted learning, a new operator for EAs introduced in earlier work. By extending with a learning rate adapting technique and a fitness comparison schema, the evolutionary algorithm

based on stochastic weighted learning can solve constrained optimization problems efficiently. The validity of the proposed algorithm was tested on a set of benchmark problems and the experimental results are very promising. From these results, we may conclude that the proposed algorithm seems to be a useful candidate for EAs.

References

1. Michalewicz, Z., Schoenauer, M.: Evolutionary Algorithm for Constrained Parameter Optimization Problems. Evolutionary Computation, 4(1) (1996) 1-32
2. Yao, X.: Evolutionary Computation: Theory and Applications. World Scientific, Singapore (1999)
3. Tan, K.C., Lim, M.H., Yao, X., Wang L.P. (eds.): Recent Advances in Simulated Evolution And Learning. World Scientific, Singapore (2004)
4. Jun, Y., Xiande, L., Lu, H.: An evolutionary algorithm based on stochastic weighted learning for continuous optimization. Proc. of 2003 IEEE International Conference on Neural Networks & Signal Processing, Nanjing, China (2003)
5. Koziel, S., Michalewicz, Z.: Evolutionary algorithms, homomorphous mapping and constrained parameter optimization. Evolutionary Computation, 7(1) (1999) 19-44
6. Runarsson, T.P., Yao, X.: Stochastic Ranking for Constrained Evolutionary Optimization, IEEE Trans. on Evolutionary Computation, Vol.4, No.3 (2000)
7. Hamida, S.B., Schoenauer, M.: ASCHEA: New results using adaptive segregational constraint handling. Proc. of the 2002 Congress on Evolutionary Computation (2002)