A New Model Based Multi-objective PSO Algorithm

Jingxuan Wei[1,2] and Yuping Wang[1]

[1] School of Computer Science and Technology, Xidian University, Xi' an 710071, China
ywang@xidian.edu.cn
[2] Department of Mathematics, Xidian University, Xi' an 710071, China
wjxjingxuan@yahoo.com.cn

Abstract. In this paper, the multi-objective optimization problem is converted into the constrained optimization problem. For the converted problem, a novel PSO algorithm with dynamical changed inertia weight is proposed. Meanwhile, in order to overcome the drawback that most algorithms take pareto dominance as selection strategy but do not use any preference information. A new selection strategy based on the constraint dominance principle is proposed. The computer simulations for four difficulty benchmark functions show that the new algorithm is able to find uniformly distributed pareto optimal solutions and is able to converge to the pareto-optimal front.

1 Introduction

The use of evolutionary algorithm for multi-objective optimization has significantly grown in the last few years, giving rise to a wide variety algorithms[1]-[4]. EMO researchers have produced some clever techniques to maintain diversity [5], new algorithm that uses very small population size [6].

Particle swarm optimization (PSO) is a recently heuristic algorithm inspired by a bird flock. PSO has been found to be successful in a wide variety fields, but until recently it had not been extended to deal with the multi-objective problems.

PSO seems suitable to deal with multiple objectives, because of its high convergence speed that the algorithm presents for single-objective optimization [7].

In this paper, we present a novel PSO algorithm which allows PSO algorithm to deal with multi-objective optimizations. Firstly, because the inertia weight ω is a very important parameter in standard version, it can control algorithm's ability of exploitation and exploration so the accumulation factor of the swarm is introduced in the new algorithm, and the inertia weight is formulated as the function of the factor. In each generation, the ω is changed dynamically according to the accumulation factor. Secondly, the multi-objective optimization problem is converted into the constrained optimization problem. Based on the converted problem, we have added a constraint-handling mechanism that can improve the exploratory capabilities of the original algorithm.

Y. Wang, Y. Cheung, and H. Liu (Eds.): CIS 2006, LNAI 4456, pp. 87–94, 2007.

2 Basic Concepts

Multi-objective optimization problems can be described as follows:

$$\min F(x) \tag{1}$$

Where, $F(x)=(f_1(x), f_2(x), \cdots f_m(x)), x\in \Omega \subset R^n$.

Definition 1: A point x^0 is said to dominate x^1 ($x^0 \prec x^1$), if for every $i\in \{1,2,\cdots m\}$ $f_i(x^0)\le f_i(x^1)$ $\wedge \exists i\in \{1,2\cdots m\}$, such that $f_i(x^0)< f_i(x^1)$.

Definition 2: A point $x^* \in \Omega$ is pareto optimal if there exists no feasible vector $x\in \Omega$, such that $x \prec x^*$.

3 Model of Multi-objective Optimization

3.1 Measure of the Quality of Solutions

Definition 3: Suppose the t-th swarm is composed of the particles $x_t^1, x_t^2 \cdots x_t^N$, let p_t^i is the number of the particles that dominate x_t^i .Then R_t^i is called the rank of particle x_t^i, $R_t^i = 1 + p_t^i$.

3.2 Measure of the Uniformity of Solutions

The aim of the multi-objective optimization is to generate a set of uniformly distributed pareto optimal solutions in the objective space. Based on this, the measure of the uniformity of solutions is given.

Definition4: Suppose the t-th swarm is composed of the particles $x_t^1, x_t^2 \cdots x_t^N$, we calculate the distances between x^i and the other particles in the objective space, and rank these distances. D_1^i and D_2^i are two smallest distances, then the crowding-distance of x^i is denoted as $crowd_i = \dfrac{D_1^i + D_2^i}{2}$. let $\overline{crowd_t} = \dfrac{1}{N}\times\sum_{i=1}^{N} crowd_i$ denote the mean value of crowding – distances of individuals and $Var_t = \dfrac{1}{N}\times \sum_{i=1}^{N} (crowd_i - \overline{crowd_t})^2$ denotes the crowding-distance variance of the t-th swarm. It can be seen that the smaller the crowding-distance variance of the t-th swarm, the more uniformity the t-th swarm.

3.3 Transform Multi-objective Optimization into the Constrained Optimization Problem

From the analysis mentioned above, it can be seen that if the ranks of all individuals are regarded as the constraints and the measure of the uniformity of solutions is regarded as the objective function, then the multi-objective optimization can be converted into the following constrained optimization problem:

$$\begin{cases} \min \quad Var_t \\ s.t. \quad R_t = 1 \end{cases} \tag{2}$$

4 Selection Operator

Most of multi-objective algorithms take pareto dominance as their selection strategy but do not use any preference information. However, these algorithms can not perform well on the problems that have many multi objectives. In order to overcome this problem, a new selection strategy for problem (2) is proposed.

4.1 If two particles are infeasible, we prefer to select the one with the smaller constraint violation, namely the one has the smaller rank.
4.2 If one particle is feasible and the other is infeasible, we prefer to select the feasible particle, namely the one has the rank one.
4.3 If two particles have the same rank, we prefer to select the one with the smaller objective value (e.g. x^i and x^j have the same rank one, then we calculate the crowding-distances of x^i and x^j in the set S based on the definition4, and choose the one with the biggest crowding-distance, the S denotes the set which is composed of all particles of rank one).The above process can distinguish the particles which locate in the sparse region and the crowded region.

5 The Accumulation Factor of the Swarm

PSO initialized the flock of birds randomly over the searching space, every bird is called a “particle”. At each generation, each particle adjusts its velocity vector, based on its best solution (pbest) and the best solution of all particles (gbest). The swarm is composed of N particles ($P_1, P_2, \cdots P_N$) , each particle’s position is represented as P_i, the velocity of this particle is denoted as V_i. At (t+1)th generation, each particle updates its position according to the following equations:

$$V_i(t+1) = \omega V_i(t) + c_1 r_1 (pbest_i(t) - P_i(t)) + c_2 r_2 (gbest(t) - P_i(t)) \tag{3}$$

$$P_i(t+1) = P_i(t) + V_i(t+1) \tag{4}$$

Where ω is the inertia weight in the range [0.1, 0.9]. c_1 and c_2 are positive constants.

One factor influences the property of algorithm is the accumulation degree of the swarm. We define $s = \frac{1}{N \cdot L} \cdot \sum_{i=1}^{N} \sqrt{\sum_{d=1}^{n} (p_{id} - \overline{p_d})^2} \in (0,1)$, where N is the population size, n is the number of variables, L is the length of the maximum diagonal in the search space, p_{id} indicates the dth coordinate of the ith particle, $\overline{p_d}$ indicates the average values of all particles in the dth coordinate . The smaller the value of s, the more centralized the swarm is. When the swarm is centralized, it becomes difficult for the algorithm to break away from the local optimum.

If the particles are sparse, the swarm is not easy to plunge into the local optimum. But when particles are centralized, it becomes easy to plunge into the local optimum.

From above, we know that ω will increase when particles are centralized, so ω can be described as follows:

$$\omega = \omega_0 - s\omega_s \tag{5}$$

Where, $\omega_0 = 1$, $\omega_s \in (0.1, 0.2)$

6 The Proposed Algorithm

Step1: Given swarm size N. Generate the initial swarm P(t) randomly, and copy non-dominated members of P(t) to $\overline{P}$ set t=1.
Step2: initialize the memory of every particle (this memory serves as a guide to travel through the search space). For $i = 1\ to\ N$, $pbest_i(t) = P_i(t)$, $P_i(t)$ indicates the i-th particle in P(t).
Step3: initialize the velocity of every particle, let $V_i(t) = 0$.
Step4: (a) compute the new speed of each particle using the expression (3): where $gbest(t)$ is taken from $\overline{P}$. Firstly, we compute the crowding-distances of all particles in $\overline{P}$, and choose the one with the biggest crowding-distance as $gbest(t)$.
(b) compute the new positions of the particles adding the speed by using the expression (4). The new swarm is defined as $P'(t+1)$.
(c) copy the nondominated members of $P'(t+1)$ to $\overline{P}$, and remove the dominated members from $\overline{P}$.After that choose N members from $P(t) \cup P'(t+1)$ to constitute the next swarm $P(t+1)$.In this study, selection operator in section 4 is used to choose the N members . Set t=t+1.
(d) when the current position of the particle is better than the position contained in its memory, the particle's position is updated using $pbest_i(t) = P_i(t)$.The criterion to decide what position from memory should be retained is simply to apply the pareto dominance.

Step5: loop to step4 until a stopping criterion is met, usually a given maximum generations.

7 Simulation Results

To evaluate the efficiency of the new algorithm NMPSO, we choose four benchmark functions [8]. All experiments were performed in matlab. The parameter is described as follows: swarm size N=100, r_1 and r_2 are the random numbers in [0, 1], c_1 and c_2 are positive constants. n is the number of the decision variables. Number of generations: 250.

7.1 Test Functions

Each of the test functions defined below is structured in the same manner :

$$\begin{aligned}
&\min \quad F(X) = (f_1(x_1), f_2(X)) \\
&s.t. \quad f_2(X) = g(X)h(f_1(x_1), g(X)) \\
&where \quad X = (x_1, x_2 \cdots x_n)
\end{aligned}$$

$$F_1 \ : f_1(x_1) = x_1$$

$$g(X) = 1 + 9\sum_{i=2}^{n} x_i /(n-1)$$

$$h(f_1, g) = 1 - (f_1 / g)^2$$

where n=30, $x_i \in (0,1)$, the pareto front is nonconvex.

$$F_2 : f_1(x_1) = x_1$$

$$g(X) = 1 + 9\sum_{i=2}^{n} x_i /(n-1)$$

$$h(f_1, g) = 1 - \sqrt{f_1} / g - (f_1 / g)\sin(10\pi f_1)$$

where n=30, $x_i \in (0,1)$.

$$F_3 : f_1(x_1) = x_1$$

$$g(X) = 1 + 10(n-1) + \sum_{i=2}^{n} (x_i^2 - 10\cos(4\pi x_i))$$

$$h(f_1, g) = 1 - \sqrt{f_1 / g}$$

where n=10, $x_1 \in (0,1)$, $x_2, \cdots x_n \in (-5,5)$.

$$F_4 \ : f_1(x_1) = 1 - \exp(-4x_1)\sin^6(6\pi x_1)$$

$$g(X) = 1 + 9(\sum_{i=2}^{n} x_i / (n-1))^{0.25}$$

$$h(f_1, g) = 1 - (f_1 / g)^2$$

Where n=10, $x_i \in (0,1)$

7.2 Computation Results

We execute 10 times on each test problem independently, and compare the results with the other 8 algorithms in [8]. In figure 1-4, the pareto fronts achieved by the

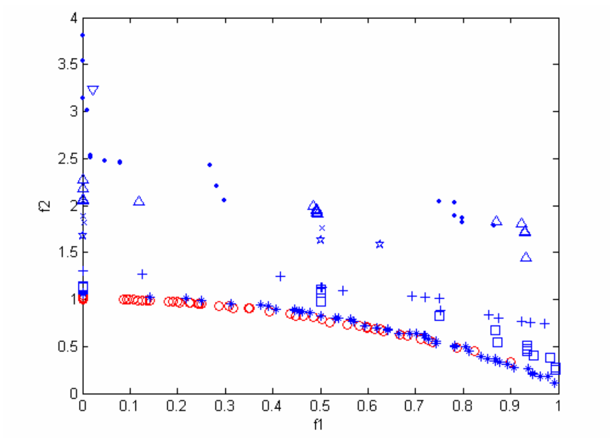

Fig. 1. Comparison results of 9 algorithms on function 1

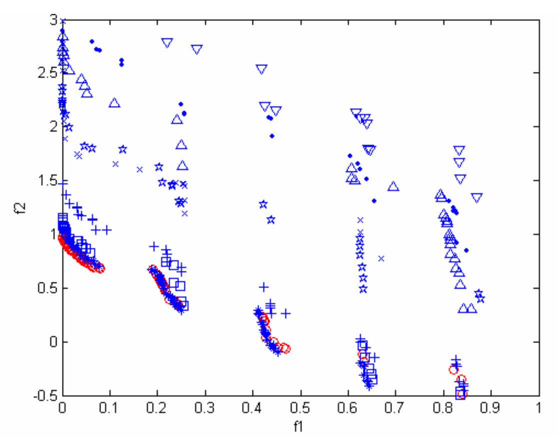

Fig. 2. Comparison results of 9 algorithms on function2

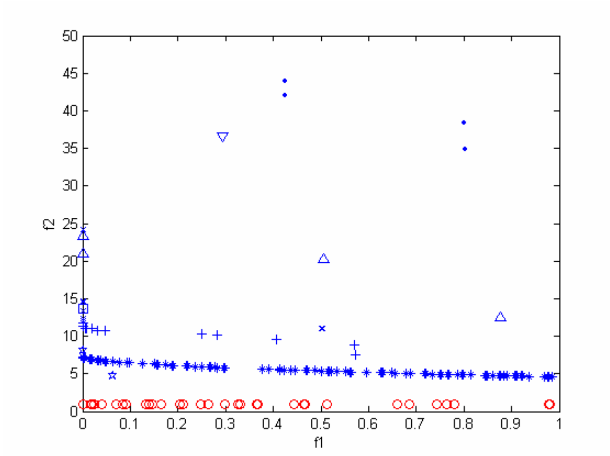

Fig. 3. Comparison results of 9 algorithms on Function 3

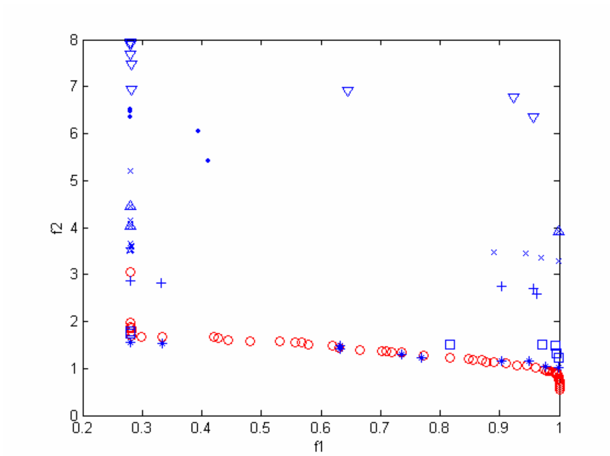

Fig. 4. Comparison results of 9 algorithms on function4

different algorithms are visualized. Per algorithm and test function, the outcomes of the first five runs were unified, and the dominated solutions were removed from the union set; the remaining points are plotted in the figures. Where ●,×,∧,+,∨,*, □, ∘,☆ denote the algorithms of Ffga, Hlga, Npga, Nsga, Rand, Spea, Soea, Nmpso and Vega.

The simulation results of the 8 algorithms in [8] are chosen from http://www.tik.ee.ethz.ch/~zitzler/testdata.html.

It can be seen from Fig1 to Fig4 that compared with the other 8 algorithms, the NMPSO can find more pareto-optimal solutions which are scattered more uniformly over the entire pareto front and the pareto front of NMPSO is in the below of the other compared pareto fronts. On average, the proposed algorithm requires 1250 function evaluations to find 100 pareto-optimal solutions.

8 Conclusions

In this paper, the multi-objective optimization problem is converted into the constrained optimization problem. For the converted problem, a novel PSO algorithm with dynamical changed inertia weight is proposed. Meanwhile, in order to overcome the drawback that most algorithms take pareto dominance as selection strategy but do not use any preference information. A new selection strategy based on the constraint dominance principle is proposed. The computer simulations for four difficulty benchmark functions show that the new algorithm is able to find uniformly distributed pareto optimal solutions and is able to converge to the pareto-optimal front.

Acknowledgements

This research is supported by National Natural Science Foundation of China (No.60374063).

References

1. Coello Coello, C.A., Van Veldhuizen, D.A., Lamont, G.B.: Evolutionary Algorithms for solving Multi-Objective Problems. Kluwer, Norwell, MA (2002)
2. Moon, P.H.A: Technique for orthogonal Frequency Division Multiplexing Offset Correction [J]. IEEE Trans on Commun 42(10), 2908–2914 (1994)
3. Zitzler, E., Laumanns, M., Thiele, L.: Spea2: Improving the Strength Pareto Evolutionary Algorithm [R]. Zurich (2001)
4. Veldhuizen, D.V.: Multiobjective Evolutionary Algorithms: Classifications, Analysis, and New Innovations [D]. Ph.D dissertation, pp. 22–24. Air University, USA (1999)
5. Deb, K., Pratap, A. et al.: A Fast and Elitist Multi-objective Genetic Algorithm: NSGAII [J]. IEEE Trans on Evolutionary Computation 6(2), 182–197 (2002)
6. Coello Coello, C.A., Pulido, G.T.: Multiobjective optimization using a micro-genetic algorithm. In: proceeding of GECCO'2001, pp. 274–282 (2001)
7. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of the IEEE International Conference on Neural Networks, vol. IV, pp. 1941–1948. IEEE Service Center, Piscataway, NJ (1995)
8. Zitzler, E., Deb, K., Thiele, L.: Comparison of Multiobjective Evolutionary Algorithms: Empirical Results [J]. Evolutionary Computation 8(2), 1–24 (2000)