Cultural Particle Swarm Algorithms for Constrained Multi-objective Optimization

Fang Gao¹, Qiang Zhao², Hongwei Liu¹, and Gang Cui¹

¹ School of Computer Science and Technology, Harbin Institute of Technology,
150001 Harbin, China

² School of traffic, Northeast forestry university, 150040 Harbin, China
gaofang@hit.edu.cn, qyangzhao@163.com,
{lhwcg}@ftcl.hit.edu.cn

Abstract. In this paper, we propose to integrate particle swarm optimization algorithm into cultural algorithms frame to develop a more efficient cultural particle swarm algorithms (CPSA) for constrained multi-objective optimization problem. In our CPSA, the population space of cultural algorithms consists of $n+1$ subswarms which are used to search for the n single-objective optimums and an additional multiobjective optimum. The belief space accepts 20% elite particles from each subswarm and further takes crossover to create Pareto optimums. Niche Pareto tournament selection is further executed to ensure Pareto set to distribute uniformly along Pareto frontier. Additional memory of Pareto optimums spool is allocated and updated in each iteration to keep resultant Pareto solutions. Besides, a direct comparison method is employed to handle constraints without needing penalty functions. Two examples are presented to demonstrate the effectiveness of the proposed algorithm.

Keywords: Multi-objective optimization, Cultural algorithms, Particle swarm optimization, Pareto optimums, Crossover.

1 Introduction

Multi-objective optimization (MOP) has assumed greater importance in many real engineering applications. A large number of algorithms based on evolutionary and swarm intelligence have been proposed for the solution of MOP during the last decades, such as NSGA, VEGA and other GA-based algorithms [1,2,3]. During the past decade, the particle swarm optimization (PSO) algorithm, proposed by James Kennedy and Russell Eberhart in 1995 [4], has gained more attraction for its simplicity and effectiveness. Recently, many researchers began to use PSO to solve multi-objective optimization problems. However, the performance of simple PSO greatly depends on its parameters, and it often suffers from the problem of being trapped in local optima that is to cause premature convergence. For example, the original PSO had difficulties in controlling the balance between exploration and exploitation because it tends to favor the intensification search around the ‘better’ solutions previously found [5,6,7]. Therefore, many approaches have proposed various kinds of approaches to improve the standard PSO algorithm. Meantime Cultural Algorithms

(CA), proposed by Reynolds [8], is a novel evolutionary computational frame based on concept of culture of human society, which shows higher intelligence in treating all kinds of complicated problems [9,10]. In this paper we propose to integrate PSO into the frame of CA and put forward a cultural particle swarm algorithm (CPSA) for constrained multi-objective optimization problems. It synthesizes both the advantages of PSO algorithm and CA and overcomes their drawbacks

2 Introduction to PSO Algorithm

The particle swarm optimization algorithm first randomly initializes a swarm of particles. Each particle is represented as $X_i = (x_{i,1}, x_{i,2}, \dots, x_{i,n})$, $i = 1, 2, \dots, N$, where N is the swarm size, and n is the total dimension number of each particle. Each particle adjusts its trajectory towards its own previous best position ***pbest*** and the previous global best position ***gbest*** attained by the whole swarm. In the k th iteration, the i th particle with respect to the j th dimension is updated by

$$v_{i,j}^{(k+1)} = v_{i,j}^{(k)} + c_1 r_1 (pbest_{i,j}^{(k)} - x_{i,j}^{(k)}) + c_2 r_2 (gbest_j^{(k)} - x_{i,j}^{(k)}), \quad (1)$$

$$x_{i,j}^{(k+1)} = x_{i,j}^{(k)} + v_{i,j}^{(k+1)}, \quad (2)$$

where $x_{i,j}^{(k)}$ and $v_{i,j}^{(k)}$ are the current position and velocity, respectively. c_1 and c_2 are acceleration constants, r_1 and r_2 are random numbers within the interval of $[0,1]$.

The procedure of the PSO algorithm for optimization can be described as follows:

- (1) Initializing: Generate N particles with random positions $X_1, X_2, \dots, X_N$, and velocities $V_1, V_2, \dots, V_N$, $i = 1, 2, \dots, N$.
- (2) Evaluate the fitness of each particle based on the problem's objective function.
- (3) Individual and global best positions updating: If $f(pbest_i) > f(X_i)$, then let $pbest_i = X_i$, and search for the minimum value $f_{\min}$ among $f(pbest_i)$, If $f(gbest) > f_{\min}$, let $gbest = X_{\min}$, $X_{\min}$ is the particle associated with $f_{\min}$.
- (4) Velocity and position updating with the Eq. (1) and (2).
- (5) Repeat Steps of (2) to (5) until a given maximum iterations number are achieved.

3 Frame of Cultural Algorithms

Cultural algorithms provide for interaction and mutual cooperation of two distinct levels of evolution: a population space and a belief space. It models cultural evolution by allowing a collection of information at a macro evolutionary level. Such information can be shared among single individuals whose evolutionary dynamics constitute the micro level. The two levels influence each other through the communication protocol. The presence of the belief space provides for a global knowledge repository and can guide the search towards better solutions, by using the knowledge to prune large portions of the state space. A framework for a CA is shown in Fig. 1.

Fig. 1. Framework of cultural algorithms

As shown in Fig.1, the two spaces are connected together by an explicit communication protocol composed of two functions: an acceptance function and an influence function. The contents of the belief space can be altered via an updating function. In the population space, individuals are first evaluated by a performance function, and then new individuals are created by a modification function. In Reynolds’s frame of cultural algorithm [8], an evolutionary planning (EP) model is employed in the population space. In this paper, multiple subswarms PSO algorithm will replace EP model for the constrained MOP problems.

4 Cultural PSO Algorithm (CPSA)

4.1 Frame of CPSA

In our dual evolution approach of CPSA, PSO algorithm is integrated into the frame of cultural algorithms for the solution of MOP as shown in Fig. 2.

The swarm intelligence of PSO is used for the evolution of population space. Since there are totally q objective functions, the population space include q subswarms, with the i th swarm evolves with $f_i(x)$ ($i=1,2,\dots,q$) as its single optimization objective respectively. Besides, another additional subswarm is also added into the population space, and this subswarm randomly selects one function from $f_1(x), f_2(x), \dots, f_q(x)$ as its single optimization objective in every cycle of k_0 iterations. After every cycle of k_0 iterations, each subswarm outputs 20% elite particles within its whole subswarm to the belief space. All the elite particles coming from different subswarms constitute the belief space and it further takes crossover operations to create Pareto optimums.

In this crossover, two particles are randomly selected as parents and performed as

$$X'_i = \alpha \cdot X_i + (1 - \alpha) \cdot X_j, \tag{3}$$

$$X'_j = (1 - \alpha) \cdot X_i + \alpha \cdot X_j, \tag{4}$$

where α is a random number extracted from region (0,1).

Fig. 2. Frame of cultural particle swarm optimization for MOP

Both the offsprings X'_i and X'_j are compared with their parent X_i and X_j . If X'_i and X'_j are nondominated, they replace their parent. Otherwise X'_i and X'_j will be discarded and their parent will be retained.

After crossover operation, a selection mechanism based on Niche Pareto tournament approach is executed to pick better particle for reproduction. The detailed selection procedure is as follows: Randomly take out two candidate particles, as well as a comparative set, which include a certain number of particles used for comparison, compare each candidate particle with the comparative set respectively, and two possible results maybe occur as:

- (1) If one candidate particle is dominated by the comparative set and another is not, then the nondominated one will be selected for reproduction.
- (2) If both the candidate particle are dominated by or dominate the comparative set, then the one with less niche count will be selected for reproduction.

Niche count in procedure (2) can be obtained by computing the sum of sharing function of the whole swarm as follow [1]:

$$m_i = \sum_{j=1}^{swarm_size} sh(d_{i,j}), \quad (5)$$

where $sh(\cdot)$ is denoted as sharing function, which takes a power law function as

$$sh(d_{i,j}) = \begin{cases} 1 - \left(\frac{d_{i,j}}{\sigma_{share}} \right)^a & \text{if } d_{i,j} < \sigma_{share} \\ 0 & \text{otherwise} \end{cases}, \quad (6)$$

where σ_{share} is denoted as niche radius, a is constant and $d_{i,j}$ is distance between particle i and j measured in decision space.

The above Niche Pareto tournament approach ensures uniformly convergence to Pareto frontier, other than undesired premature.

As mentioned above, the belief space accepted the elite particles so as to be shared with the entire population. Whereas after the updating operation of crossover and selection, the belief space will supervise the evolution of the population space by adding a correction item to its velocity by updating formula in Eq. (1) as

$$v_{i,j}^{(k+1)} = v_{i,j}^{(k)} + c_1 r_1 (pbest_{i,j}^{(k)} - x_{i,j}^{(k)}) + c_2 r_2 (gbest_j^{(k)} - x_{i,j}^{(k)}) + c_3 r_3 (Gbest_j^{(k)} - x_{i,j}^{(k)}) \quad (7)$$

where c_3 is acceleration constant. r_3 is a random number within the interval of $[0,0.5]$. $Gbest_j$ is the position attained by the elite particle in belief space. Roulette wheel selection approach is employed to select one as $Gbest_j$ from the belief space.

To keep a set of Pareto optimums in the evolution, an independent external memory called population pool is allocated to store Pareto set as seen in Fig. 2. It is updated by deleting all dominated solutions and accepting new Pareto solutions from the belief space in each generation.

4.2 Constraints Handling

For general constrained optimization problem, some infeasible individuals maybe exist near the global optimum and holds high fitness values. Although they are infeasible in current iteration, further operations maybe make them to create new feasible offspring with higher fitness value. Thus it is helpful for the optimization to keep one small part of infeasible but good individuals. Inspired by [11,12,13], a feasibility-based constraint handling rule, called direct comparison-method is employed in this paper to handle constraints, it presents the comparison rules among individuals, as well as the adaptation strategy to keep certain proportion of infeasible individual in the population. To describe the magnitude that all inequality constraints are violated, a measuring function is defined as follows:

$$viol(x) = \sum_{j=1}^m f_j(x), \quad (8)$$

where $f_j(x)$, $j=1,2,\dots,m$, is a series of penalty functions to measure what extent each constraint is violated to, it is defined as:

$$f_j(x) = \begin{cases} \max\{0, g_j(x)\} & \text{if } 1 \leq j \leq q \\ |h_j(x)| & \text{if } q+1 \leq j \leq m \end{cases} \quad (9)$$

For a predefined constant ϵ ($\epsilon > 0$), two individuals are compared and treated according to the follows handling rules:

- (1) When two individuals are both feasible, select the one with higher fitness.
- (2) When two individuals are both infeasible, select neither.
- (3) When an individual x is feasible and another individual x' is infeasible, and if $viol(x) \leq \epsilon$, compare their fitness and select the one with higher fitness.

To maintain a rational proportion of infeasible individuals in the whole population, an adaptive updating strategy for ε can be done when every cycle of k_0 generations of evolution is completed, the update is as

$$\varepsilon_{new} = \begin{cases} 1.2\varepsilon_{old} & \text{if } a/pop_size \leq p \\ 0.8\varepsilon_{old} & \text{if } a/pop_size > p, \\ \varepsilon_{old} & \text{else} \end{cases} \quad (10)$$

where a is number of infeasible individuals, p is a predefined constant to determine the proportion of infeasible individuals.

For the two offsprings created by Eqs. (3) and (4) in the belief space, the above constraints handling rule can be directly used. For the particle updating formula in Eq. (1), the above handling rule can be executed as follows: suppose that $pbest^{(k)}$ represents $pbest$ of the i th particle at generation k , and $x_i^{(k+1)}$ represents the newly generated position of the i th particle at generation $k+1$. $pbest^{(k)}$ will be replaced by $x_i^{(k+1)}$ at any of the following occasions:

- (1) $pbest^{(k)}$ is infeasible, but $x_i^{(k+1)}$ is feasible.
- (2) Both $pbest^{(k)}$ and $x_i^{(k+1)}$ are feasible, but $f(x_i^{(k+1)}) < f(pbest^{(k)})$.
- (3) Both $pbest^{(k)}$ and $x_i^{(k+1)}$ are infeasible, but $viol(x_i^{(k+1)}) < viol(pbest^{(k)})$. Similarly, $gbest$ is updated based on the above rule at every generation.

4.3 Procedure of CPSA

Our CPSA actually adopts a dual evolution mechanism. Each subswarm in the population space and the elite swarm in the belief space can evolve synchronously with a parallel multi-thread mode in one computer or multi-computer c/s network respectively to increase computing speed. However single-thread basic procedure of the CPSA can be described as follows:

```

Begin
t=0;
Initialize Population Space POP(t) and Belief Space BLF(t);
repeat
    repeat //adjust Population Space
        For i=1 to n+1
            Evaluate subswarm i;
            Individual and global best positions updating
                for subswarm i;
            Velocity and position updating for subswarm i;
        End
    Until expected number of iterations achieved.
    //Adjust (BLF(t),Accept(POP(t)));
    Select 20% elite particles to belief space
    Crossover operation in belief space
    // Adjust(BLF(t));
    Update Pareto solutions set
    Niche Pareto tournament selection

```

```
Update population pool
Output Gbesti to each subswarm //Influence(POP(t));
Until termination condition achieved
End
```

5 Numerical Example

Two simple and typical single-variable and double-variable double-objective problem is first presented to demonstrate the proposed algorithm. It is described as follows:

- (1) Example 1 : $\min \{ f_1 = x^2; f_2 = (x-2)^2 \text{ s. t. } x \in R^1 \}$
- (2) Example 2: $f_1(x) = -25(x_1 - 2)^2 - (x_2 - 2)^2 - (x_3 - 1)^2 - (x_4 - 1)^2 - (x_5 - 1)^2$
 $f_2(x) = (x_1 - 1)^2 + (x_2 - 1)^2 + (x_3 - 1)^2 + (x_4 - 1)^2 - (x_5 - 1)^2$
s. t. $\{ g_1(x) = x_1 + x_2 - 2 \geq 0; g_2(x) = 6 - x_1 - x_2 \geq 0; g_3(x) = 2 + x_1 - x_2 \geq 0;$
 $g_4(x) = 2 - x_1 + 3x_2 \geq 0; g_5(x) = 4 - (x_3 - 3)^2 - x_4 \geq 0;$
 $g_6(x) = (x_5 - 3)^2 + x_6 - 4 \geq 0; 10 \geq x_i \geq 0, i = 1,2,3,4,5,6 \}$

Fig.3 presents the curves of objective functions f1(x) and f2(x) with decision variable x for example 1. Using the proposed algorithm, its optimums in criterion space are shown in Fig.4 and Fig.5 after 100 and 400 generations of iterations. The Pareto optimums are shown in Fig. 6 for example 2.

Fig. 3. f₁(x) and f₂(x) with x

Fig. 4. f₁(x) and f₂(x) after 100 generations

Fig. 5. f₁(x) and f₂(x) after 400 generations

Fig.6. f₁(x) and f₂(x) of example 2

6 Conclusion

This paper has proposed a novel dual-evolutionary cultural particle swarm algorithm for constrained multi-objective optimization problem. Multiple particle subswarms make up of population space of cultural algorithms frame. These subswarms ensure to find each single-objective optimum, as well as a multi-objective solution. The belief space accepts a certain number of elite particles and takes crossover operation for Pareto optimums. Additional Nicho Pareto tournament selection can make Pareto set to be uniformly distributed in Pareto frontier. A direct comparison method for constraint handling overcomes the disadvantages of penalty function methods.

Simulation results showed that our proposed CPSA is of good performances, and more typical function test will be discussed in future.

References

1. Gen, M., Cheng R.: Genetic Algorithms and Engineering Optimization. John Wiley & Sons, Inc. (2000)
2. Schaffer, J.D.: Multiple objective optimization with vector evaluated genetic algorithms, in: Proceedings of an International Conference on Genetic Algorithms and Their Applications, Pittsburgh, PA, (1985)93–100
3. Srinivas, N., Deb, K.: Multiobjective optimization using nondominated sorting in genetic algorithms, *Evolutionary Computation* 2 (3) (1994) 221–248
4. Kennedy, J. E., Eberhart, R.: Particle Swarm Optimization. In: Proceedings of the IEEE International Conference on Neural Networks, Vol. 4. Perth. Australia. (1995) 1942–1948
5. Silva, A., Neves, A., and Costa, E.: An Empirical Comparison of Particle Swarm and Predator Prey Optimization. *Lecture Notes Comput. Sci.*, 2464(2002) 103–110
6. Schutte, J. F., Groenword, A. A.: A Study of Global Optimization Using Particle Swarms. *J. Global Optimiz.*, 31(2005) 93–108
7. Ho, S. L., Yang, S. Y., Ni, G. Z., and Wong, H. C.: A Particle Swarm Optimization Method with Enhanced Global Search Ability for Design Optimizations of Electromagnetic Devices. *IEEE Transactions on Magnetics*, 42(2006) 1107–1110
8. Reynolds, R. G.: An Introduction to Cultural Algorithms. Proceedings of the 3rd Annual Conference on Evolutionary Programming, World Scientific Publishing, (1994) 108–121
9. Jin, X.D., Reynolds, R. G.: Using Knowledge-Based Evolutionary Computation to Solve Nonlinear Constraint Optimization Problems: a Cultural Algorithm Approach, *IEEE*, (1999)1672–1678
10. Jin, X.D., Reynolds, R. G.: Mining Knowledge in Large Scale Databases Using Cultural Algorithms with Constraint Handling Mechanisms. Proceeding of the 2000 Congress on Evolutionary Computation., *IEEE*, (2000) 1498–1506
11. Li, M. Q., Kou, J. Z., Lin, D., Li, S. Q.: Based Theory and Application of Genetic algorithm, 3rd edition. Science Press, (2004)
12. Deb, K.: An Efficient Constraint Handling Method for Genetic Algorithms. *Comput. Meth. Appl. Mech. Eng.*, 186 (2000) 311–338
13. He, Q., Wang, L.: A Hybrid Particle Swarm Optimization with a Feasibility-based Rule for Constrained Optimization, *Applied Mathematics and Computation*, (2006), doi:10.1016/j.amc.(2006) 134