A Revised Particle Swarm Optimization Approach for Multi-objective and Multi-constraint Optimization

JI Chunlin

School of Information Science and Engineering, Northeastern University,
ShenYang 110004, China
ji_chunlin@hotmail.com

Abstract. Many real world design or decision-making problems involve simultaneous optimization of multiple objectives, while satisfying multiple constraints. In this paper, some novel adaptations were given to the recent bio-inspired optimization approach, Particle Swarm Optimization (PSO), to form a suitable algorithm for these multi-objective and multi-constraint optimization problems. Divided Range Multiobjective Particle Swarm Optimization (DRMPSO) was presented, extending PSO for distributed computing. Inspired by the biological phenomenon of symbiosis, a problem-independent constraint handling technique was created, by introducing symbiosis mechanism to PSO, to deal with the multiple constraints. The proposed algorithm was tested on three benchmark problems, comparing with two other approaches in an efficient comparison form.

1 Introduction

Many real-world design optimizations encountered in industrial electronics always involve simultaneous optimization of multiple objectives, while satisfying multiple constraints. Consider the following general multi-objective multi-constraint optimization problem:

$$\begin{cases} \text{Minimize } \bar{f}(\vec{x}) = [f_1(\vec{x}), f_2(\vec{x}), \cdots, f_k(\vec{x})] \\ \text{subject to } g_j(\vec{x}) \le 0, \quad j = 1,2,\cdots,m \end{cases} \tag{1}$$

where $\vec{x} = \{x_1, \cdots, x_n\}$ is a n dimensional vector, each $x_i (i = 1, \cdots, n, n \ge 1)$ can be real-valued, integer-valued or boolean-valued. $f_i(\vec{x})$ $(i = 1, \cdots, k, k \ge 1)$, $g_j(\vec{x})$ $(j = 1, \cdots, m, m \ge 0)$ are linear or nonlinear arbitrary functions, and no constraint is assumed when $m = 0$.

In this paper, I will show a revised Particle Swarm Optimization to find multiple nondominated solutions to the multi-objective and multi-constraint optimization problem, in which a problem-independent technique for constraint satisfaction is introduced.

2 Particle Swarm Optimization

Particle Swarm Optimization (PSO) was first proposed by Eberhart and Kennedy [1], which was inspired by the choreography of a bird flock and can be seen as a distributed behavior algorithm that performs multidimensional search. According to PSO, the behavior of each individual is affected by either the best local or the best global individual to help it fly through a hyperspace. Moreover, an individual can learn from its past experiences to adjust its flying speed and direction. Therefore, by observing the behavior of the flock and memorizing their flying histories, all the individuals in the swarm can quickly converge to near-optimal geographical positions.

However, due to PSO's single-point-centered characteristic, it is difficult to locate the non-dominated points on the Pareto front since there will be more than one criteria exist to direct the velocity and position of an individual. Thus, basic version PSO algorithms are not effective to deal with multiobjective optimization. Moreover, there is no strategy in PSO to handle multi-constraints. So, in order to overcome these drawbacks and create an efficient algorithm for the multi-objective and multiconstraint optimization problem, in this work, the PSO algorithm is revised as follows.

3 Distributed Multiobjective Particle Swarm Optimization

Multiobjective particle swarm optimization for MOPs, is an active field of recent research [2-4]. In this work, I give a framework of a novel model of PSO, Divided Range Multiobjective Particle Swarm Optimization (DRMPSO) for distributed computing. In the DRMPSO, the individuals are divided into sub-populations by the values of their objective function. Therefore, the efficient search can be performed and the adequate local search also carried out. This model is suitable for parallel processing. The flow of Divided Range Multiobjective Particle Swarm Optimization algorithm is explained as follows.

Step 1. Initial population (population size is N) is produced randomly. Calculate the fitness value of each individual.

Step 2. The individuals are sorted by the values of focused objective function f_i. This focused objective function f_i is chosen in turn, and turned with the loop. Then, the individuals of number N/m are chosen in accordance with the value of this focused objective function f_i. As the result, there exist m sub-populations.

Step 3. In each sub-population, one multiobjective PSO (e.g. MOPSO in [2]) has been performed for some iterations. At the end of each generation, the terminal condition is examined and the process is terminated when the condition is satisfied. When the terminal condition is not satisfied the process progresses into the next step.

Step 4. After the multiobjective optimization has been performed for k generations, all of the individuals are gathered (virtually). Then the process is going back to Step 2. This generation k is called the *sort interval*.

4 Constraint Handling Technique

There are some studies reported in literature that extended PSO to constrained optimization problems. Various constraint-handling techniques were employed to facilitate the optimization process [5-7]. In this section, I propose a new constraint handling technique for PSO inspired by biological phenomenon of symbiosis in Nature.

The PSO algorithms described so far, at least implicitly, referred to a single population (or species) searching for solutions to a single problem. But, in Nature many organisms symbiose with other organisms, that may, but does not necessarily, benefit each member. Inspired by this phenomenon, I intend to introduce symbiosis mechanism to the previous PSO. As stated before, PSO simulates the social behaviors of bird flocking. Symbiosis mechanism can bring cooperation and interdependence between different species, thus enhance the PSO. So, it is a potential research field, which needs a great deal of relevant and rigorous study. In this work, a constraint handling technique is created for PSO using symbiosis mechanism.

To handle constraints, and simultaneously, drive the population to the Pareto front, I make the following modifications to PSO:

1) Populations are composed of feasible and infeasible individuals. Unlike the symbiosis in Nature, here, feasible individual and infeasible individual may change to each other, through the velocity and position update.
2) Feasible individuals evolve towards Pareto front guided by objective functions, using one multiobjective strategy.
3) Infeasible individuals evolve towards feasibility guided by an unfeasibility evaluation function, generally the constrained function or the weighted constrained functions.
4) There is a threshold λ for the proportion of infeasible individuals in the population. At the end of each generation, check whether the proportion of infeasible individuals exceeds the threshold. If so, choose the redundant infeasible individuals and update their velocities and positions iteratively until they become feasible individuals. In this case, infeasible individual can share the information from the best global feasible individual to become feasible quickly.
5) Moreover, the threshold reduces gradually, like the "temperature" decreases in Simulated Annealing algorithms.

By the symbiosis mechanism, infeasible individuals can be allowed to evolve with feasible individuals. Breeding between feasible and infeasible solution, the population move towards the Pareto front, simultaneously try to minimize the violation of infeasible solution. Finally we will get a set of Pareto optimal solution, while the constraints are satisfied.

5 Experiments and Results

Under this multiobjective environment, the number of non-dominated solutions is directly linked to the population size, so a larger population size is preferred, e.g.

400~1000, in this work. The population is divided into four sub-populations running in four processors parallelly. Sort interval k is set to 5. The infeasible individuals proportion threshold λ gradually decreases from an initial value (e.g. 40%) approaching zero, according to the rule: $\lambda_{i+1} = \alpha\lambda_i$, for some $\alpha < 1$ (say, $0.9 < \alpha < 1$). The other parameters are set as follows: the maximum generation is 1,000; the inertia weight gradually decreases from 1.2 towards 0.1; the learning rates are both 2.0; the maximum velocity set to the dynamic range of the particle on each dimension.

The revised PSO algorithm is tested on the following three benchmark problems, which were studied extensively in [8-11]. All test problems are converted to the formulation (1) where constraints are expressed by inequalities.

Test Problem 1:

$$\begin{cases} \text{Minimine: } f_1(x) = 2 + (x_1 - 2)^2 + (x_2 - 1)^2 \\ \qquad\qquad f_2(x) = 9x_1 - (x_2 - 1)^2, \\ \text{subject to:} \\ \qquad x_1^2 + x_2^2 \le 225 \\ \qquad x_1 - 3x_2 + 10 \le 0 \\ \qquad -20 \le x_i \le 20, \forall i = 1,2 \end{cases}$$

Test Problem 2:

$$\begin{cases} \text{Minimine: } f_1(x) = x_1 \\ \qquad\qquad f_2(x) = x_2, \\ \text{subject to:} \\ \qquad x_1^2 + x_2^2 - 1 - 0.1\cos\left(16\arctan\frac{x}{y}\right) \ge 0 \\ \qquad (x_1 - 0.5)^2 + (x_2 - 0.5)^2 \le 0.5 \\ \qquad 0 \le x_i \le \pi, \forall i = 1,2 \end{cases}$$

Test Problem 3:

$$\begin{cases} \text{Minimine: } f_1(x) = -[25(x_1 - 2)^2 + (x_2 - 2)^2 + (x_3 - 1)^2 \\ \qquad\qquad + (x_4 - 4)^2 + (x_5 - 1)^2] \\ \qquad\qquad f_2(x) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2, \\ \text{subject to:} \\ \qquad x_1 + x_2 - 2 \ge 0 \\ \qquad x_1 + x_2 - 6 \le 0 \\ \qquad x_1 - x_2 + 2 \ge 0 \\ \qquad x_1 - 3x_2 - 2 \le 0 \\ \qquad (x_3 - 3)^2 + x_4 - 4 \le 0 \\ \qquad (x_5 - 3)^2 + x_6 - 4 \le 0 \\ \qquad 0 \le x_1, x_2, x_6 \le 10,\ 1 \le x_3, x_5 \le 5,\ 0 \le x_4 \le 6 \end{cases}$$

The PSO algorithm is implemented in a few lines of computer code. It requires only primitive mathematical operators, and is computationally inexpensive in terms of both memory requirements and speed. In all these test problems, convergence to a Pareto front is achieved, in some runs, by less than 100 runs, after which the improvement is very marginal. There is obvious advantage of running the PSO in 40 runs, and no marked gain beyond 80 runs. So, in the absence of a convergence criterion, it is a waste to run for a pre-determined and fixed number of runs.

Proper comparison of results from two multiobjective optimizers is a complex matter. In this work, I employ an efficient performance comparison method, given by Knowles and Corne [12]. The results of the comparison is presented in the form of a pair [a, b], where a gives the percentage of the space on which algorithm A is found statistically superior to B, and b gives the similar percentage for algorithm B. Typically, if both A and B are 'good', $a+b<100$. The result $100-(a+b)$ then, gives the percentage of the space on which results were statically inconclusive. DRMPSO is compared against two evolutionary multiobjective optimization algorithms, MOBES [8] and NSGA [10], which were applied to such kind of above test problems before. Table 1 shows the comparison of my results with results by previous approaches in this form.

Table 1. Proposed PSO vs MOBES and NSGA comparisons in three test problems

Test Problem	Proposed PSO vs MOBES in [8]	Proposed PSO vs NSGA in [10]
No. 1	[58.6, 32.8]	[59.6, 40.8]
No. 2	[52.9, 35.2]	[48.7, 28.9]
No. 3	[42.6, 29.3]	[50.3, 31.4]

6 Conclusions

The main focus of this work has been to combine a problem-independent constraint handling technique with a distributed multiobjective particle swarm optimization algorithm to find multiple nondominated solutions for the constrained multiobjective optimization problems. Multiobjective PSO algorithms had been studied extensively for multiobjective optimization problems, but they were all implemented centralized. In this paper, I gave a framework of a novel model of PSO, Divided Range Multiobjective Particle Swarm Optimization for multiobjective optimization in a distributed computing manner. To the best of my knowledge, this is the first attempt to implement PSO in distributed model.

In order to deal with the constraints, I create a problem-independent constraint handling technique for PSO inspired by biological phenomenon of symbiosis in Nature. Symbiosis mechanism is introduced to the previous PSO, bring cooperation and interdependence between different species, and thus enhance the PSO. Under the symbiosis mechanism, infeasible individuals can be allowed to evolve with feasible individuals. Breeding between feasible and infeasible solution, the population move towards the Pareto front, simultaneously try to minimize the violation of infeasible solution. Finally we will get a set of Pareto optimal solution, while the constraints are satisfied. As far as the authors know, it seems to be a new and potential research field to introduce symbiosis mechanism to PSO, which needs a great deal of relevant and rigorous study e.g. the information sharing mechanism between different species, infeasible population control schedule, and so on.

The proposed PSO algorithm was tested on three benchmark problems. Experiments show that the population converges to the Pareto front quickly. Moreover, an efficient performance comparison method was employed to compare with another two evolutionary approaches.

References

1. R. C. Eberhart and J. Kennedy, “A new optimizer using particle swarm theory”, in Proceedings of the Sixth International Symposium on Micromachine and Human Science, Nagoya, Japan (1995) 39-43
2. C. A. C. Coello and M. S. Lechuga, “MOPSO: a proposal for multiple objective particle swarm optimization”, in Proceedings of the IEEE Congress on Evolutionary Computation, Honolulu, Hawaii, USA (2002)
3. X. Hu and R. C. Eberhart, “Multiobjective optimization using dynamic neighborhood particle swarm optimization”, in Proceedings of the IEEE Congress on Evolutionary Computation, Honolulu, Hawaii, USA (2002) 1677-1681
4. K. E. Parsopoulos and M. N Vrahatis, “Particle swarm optimization method in multiobjective problems”, in Proceedings of the ACM Symposium on Applied Computing 2002, Madrid, Spain (2002) 603-607
5. K. E. Parsopoulos, and M. N. Vrahatis, “Particle swarm optimization method for constrained optimization problems”, in Proceedings of the Euro-International Symposium on Computational Intelligence 2002 (2002)
6. T. Ray, and K. M. Liew, “A swarm with an effective information sharing mechanism for unconstrained and constrained single objective optimization problem”, in Proceedings of IEEE Congress on Evolutionary Computation, Seoul, Korea (2001) 75-80
7. X. Hu, R. C. Eberhart, and Y. Shi, “Engineering optimization with particle swarm”, in Proceedings of the IEEE Swarm Intelligence Symposium 2003, Indianapolis, Indiana, USA (2003) 53-57
8. T. Binh, and U. Korn, “MOBES: A multiobjective evolution strategy for constrained optimization problems”, in Proceedings of the third international Conference on Genetic Algorithms, Brno, Czech Republic (1997) 176-182
9. T. Binh, “A multiobjective evolutionary algorithm: The study cases”, In Technical report, Institute for Automation and Communication, Barleben, Germany (1999)
10. N. Srinivas and K. Deb, “Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms”, Evolutionary Computation, Vol.2, No.3, Fall (1994) 221-248
11. D. A. Veldhuizen. “Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations”, PhD thesis, Department of Electrical and Computer Engineering. Air Force Institute of Technology, Wright-Patterson AFB, Ohio, May (1999)
12. J. D. Knowles and D. W. Corne, “Approximating the nondominated front using the Pareto archived evolution strategy”, Evolutionary Computation, Vol. 8, No. 2 (2000) 149-172