An effective hybrid search mode for multi-objective optimization with constraints

Yujun Chen[1,*], and *Wenqiang* Yuan[2]

[1]Institute of Telecommunication and Navigation Satellites, China Academy of Space Technology, Beijing 100094, China
[2]Computer College, Hangzhou Dianzi University, Hangzhou 310000, China

Abstract. In this paper a new search strategy for multi-objective optimization (MOO) with constraints is proposed based on a hybrid search mode (HSM). The search processes for feasible solutions and optimal solutions are executed in a mixed way for the existing methods. With regard to HSM, a hybrid search mode is proposed, which consists of two processes: Feasibility search mode (FSM) and optimal search mode (OSM). The executions of these two search modes are independent relatively and also adjusted according to the population distribution. In the early stage, FSM plays the leading role for exploring the feasible space since most of the individuals are infeasible. With the increase of the feasible individuals, OSM is the primary operation for the search of optimal individuals. The proposed method is simple to implement and need few extra parameter tuning. The handing method of constraints is tested on several multi-objective optimization problems with constraints. The remarkable results demonstrate its effectiveness and good performance.

Keywords: Multi-objective optimization, Hybrid search mode, Feasibility search mode, Optimal search mode.

1 Introduction

Multi-objective and multi-disciplinary design optimization problems are common exiting in design of spacecraft and other complex engineering system. The research of efficient multi-objective optimization strategy is in urgent need. Evolutionary algorithms (EA) have been applied to solve optimization problems in the fields of science research and real engineering problems including the cases with some conflicting objectives, a lot of related achievements have been obtained [1]. Many evolutionary approaches for multi-objective optimization problems are studied and a historical review of evolutionary can be seen. These involved approaches are mainly designed to solve unconstrained optimization problems [2]-[5]. In addition, researchers have also enriched these algorithms to solve multi-objective optimization (MOO) problems with constraints [6]-[7]. The challenges in the constrained optimization problems stem from the various limits on the decision variables, the various constraints, the interferences among constraints and coupling

* Corresponding author: chenyj1001@163.com

relationships between the objectives and constraints. Nevertheless, little research has been done in the field of multi-objective optimization problems with constraints in early. Multi-objective optimization problem with constraints is mathematically formulated as following.

$$\begin{aligned} &Minimize: f_i(x) = f_i(x_1, x_2, ..., x_n), i = 1, ..., k \\ &s.t.: \begin{cases} g_j(x) = g_j(x_1, x_2, ..., x_n), j = 1, ..., m \\ h_j(x) = h_i(x_1, x_2, ..., x_n), j = 1, ..., p \\ x_j^{\min} \le x_j \le x_j^{\max}, j = 1, 2, ..., n \end{cases} \end{aligned} \tag{1}$$

Herein, there are k in all of optimization objectives that need be solved for minimization simultaneously. Each optimization objective is defined in the search space R^n. The search space is restricted within m inequality and p equality constraint conditions.

In recent years, some research is conducted in constraint handling of multi-objective optimization. J. Y. Wei has proposed a new kind of constraint handing method using a new transformation strategy of constraint functions to be another objective function [8]. A. Mani has proposed a hybrid constraint handing technique for two population adaptive co-evolutionary algorithm using a self-determinative and regulating penalty factor as well as feasibility rules [9]. L. Omeltschuk proposes a generic and hybrid constraint handling scheme for particle swarm optimization [10]. H.B. Zhang has proposed an efficient constraint handling method with integrated differential evolution (IDE) which is applied in several engineering optimization [11]. C.J Zhang has proposed a backtracking search algorithm with three constraint handling methods [12]. G.F. Mario conducts the constraint handling of MOO in the application of protein structure prediction [13]. LZ Liu has proposed some generic constraints handling techniques, one of which is specifically proposed for discrete constraint optimization problems [14]. However, there are still some deficiencies: I) The transformation function is difficult to construct and also its performance cannot be ensured; II) The existing methods require fine tuning of one or more parameters, which in itself becomes an optimization problem; III) The search strategy cannot ensure that the individuals are located in the feasible region since it's a mixture strategy of feasibility and optimal [15].

In this study, a new hybrid search mode is proposed for solving multi-objective optimization problems with constraints. Firstly, FSM is launched and then OSM is done according to the distribution of feasible individuals.

2 Hybrid search strategy design

HSM adopts a flexible handling strategy for the search process according to the population distribution situation (PDS) of the evolving individuals in each iteration. Here, the percentage of infeasible individuals in the whole population p_{inf} is taken as the metric to classify different PDSs. Two thresholds are used to distinguish three PDSs: $HIGH\_INF\_S$ and LOW_INF_S. These thresholds are to indicate the percentages of infeasible individuals in the whole population are rather large and small respectively, the parameter setting of $HIGH\_INF\_S$ is close to 1.0 while it's close to 0 for LOW_INF_S. The ideal state is that eventually all of the individuals are located in the feasible region. Based on this evolution trend, three PDSs are defined as follows distinguished by the value of p_{inf}.

PDS 1: $HIGH\_INF\_S < p_{inf} \le 1$

In the initial and early phase of the evolution process, most of the individuals are infeasible and p_{inf} belongs to this interval. Particularly, the case that p_{inf} equals 1 shows no feasible individuals exist at this moment. If the given optimization problem is loosely constrained most of the individuals randomly generated in the initial population may be located in the feasible search space, which implies p_{inf} is not too large, thus the population will not be located in PDS 1.

$$\text{PDS 2:} \quad LOW\_INF\_S \le p_{inf} \le HIGH\_INF\_S$$

In this middle phase, the quantity difference of the infeasible individuals and feasible individuals is relatively small. The population presents this distribution situation during most of the evolution process.

$$\text{PDS 3:} \quad 0 \le p_{inf} < LOW\_INF\_S$$

In this phase p_{inf} is pretty small and most of the individuals are evolved into the feasible region. Particularly, the case that the p_{inf} is equal to 0 implies that all of the individuals are feasible.

The defined PDSs present the following characteristics respectively.

1) For PDS 1 none or few feasible individuals exist. Therefore, it makes little sense to execute the operation of optimal search taking advantage of the few feasible individuals. The key task is to explore the whole design space and find the feasible individuals as many as possible in the first place. Only FSM is executed, this will reduce the complexity of the evolution process during these generations.

2) With regard to PDS 2 the population is composed of some feasible individuals and some infeasible individuals. Their quantities difference is not extremely obvious. Therefore, both of the explorations for the feasible solutions using the infeasible individuals and the optimal solutions using the existing feasible individuals in the population should be conducted. The strategy in PDS2 is the most concerned focus, which makes the largest senses on the eventual search results.

3) For PDS 3 most of the infeasible individuals are evolved to be feasible and none or few infeasible individuals exist. FSM is not executed based on the following two reasons: I) The existing feasible individuals in the population is adequate enough to support the optimal exploration; II) It's a hard work to evolve the left few infeasible individuals into feasible regions.

Algorithm 1 shows the pseudo code of the proposed whole method process. The overall workflow of this method is controlled by the metric p_{inf}. p_{inf} is not an extra imported parameter but an existing state quantity that the population currently presents. According to the value of p_{inf} the corresponding handling strategy is adopted. The overall principle of the proposed strategy is logically simple: I) p_{inf} belongs to PDS 1, then only FSM is launched to find more feasible individuals as many as possible; II) p_{inf} belongs to PDS 2, this period takes the most of the whole evolution period and plays the key role on the final optimization result. HSM is executed to explore the infeasible and feasible regions in a parallel way. The infeasible individuals are gradually becoming feasible through FSM and more optimal individuals are found through the OSM; III) p_{inf} belongs to PDS 3, the key task is to find more optimal solutions as many as possible and only the OSM is launched.

Algorithm 1 Pseudo code of the whole method process

1. Initialize the population;

2. Calculate p_{inf} ;
Generation = 0;
While (generation < MaxGeneration)
If p_{inf} belongs to PDS 1
3. Execute FSM;
Else if p_{inf} belongs to PDS 2
4. Execute HSM;
Else if p_{inf} belongs to PDS 3
5. Execute OSM;
6. Calculate p_{inf} ;
End while
Return elitism;

3 Description of FSM

For the cases of PDS 1 and PDS 2 the FSM is launched. Only the sub-population of infeasible individuals is considered in FSM and the unique goal is to find more feasible spaces, which means the infeasible individuals are evolved into feasible spaces to be feasible individuals, thus, the constraint handling is conducted in this mode.

Three primary problems should be handled during this search process: I) How to deal with the constraints of the infeasible individuals reasonably; II) How to compute the fitness value of the infeasible individuals; III) How to improve the search efficiency in the infeasible regions.

3.1 PSEDO code of FSM process

Algorithm 2 shows the pseudo code of FSM. The infeasible sub-population is taken as the processed objects. The infeasible sub-population is initialized and Pinf is calculated (Line 1 and 2). The constraints on one infeasible individual are normalized (Line 3). Based on the normalized constraints the fitness of one individual is calculated (Line 4). The individuals in this sub-population are ranked (Line 5). Two infeasible individuals with good fitness are selected (Line 6). The mutation, crossover and selection operations are executed respectively (Line 7, 8 and 9). Two sub-populations are adjusted because of the new individual (Line 10) and re-calculation of p_{inf} (Line 11).

Algorithm 2 The pseudo code of feasibility search mode

1. Initialize infeasible sub-population;
2. Calculate p_{inf} ;

While (p_{inf} does not belong to PDS 3)

3. Normalize constraint values for each infeasible individual;
4. Calculate fitness of each individual using normalized constraint values;
5. Rank infeasible sub-population according to their fitness;
6. Select two infeasible individuals with good fitness;
7. Execute mutation operation for current individual using selected two individuals;
8. Execute crossover operation;
9. Execute selection operation;
10. Adjust two sub-populations;
11. Recalculate p_{inf} ;

End while

3.2 Strategy of constraint

In general, the task of constraint handling can be decomposed into three processes for a given MOO problem: I) Solve the appended constraint conditions; II) Compute the fitness value of each infeasible individual using a uniform formula; III) Rank the infeasible sub-population.

Here the involved symbols are listed: N is the population size, N_{inf} is the sub-population size for infeasible individuals, N_f is the sub-population size for feasible individuals. N equals the sum of N_{inf} and N_f .

In this work, the uniform normalization process for the constraint function and objective function is adopted as the primary strategy to solve the appended constraint conditions, in which the effect of an individual's constraint violation into its objective function is considered. The minimum and maximum values of each objective function in the population are obtained.

$$f_{\min}^{i} = \min_{x} f_i(x) \tag{2a}$$

And

$$f_{\max}^{i} = \max_{x} f_i(x) \tag{2b}$$

Using the above two values normalize each objective function i for every individual x.

$$\overline{f}^{i}_{\max} = \frac{f_i(x) - f_{\min}^{i}}{f_{\max}^{i} - f_{\min}^{i}} \tag{3}$$

where $\overline{f}^{i}_{\max}$ is the normalized value of the ith objective value of individual x.

Similarly, with regard to the constraint violation v(x) of individual x is computed as the sum the normalized violations of each constraint divided by the number of constraints.

$$v(x) = \frac{1}{m+p} \sum_{j=1}^{m+p} \frac{c_j(x)}{c_{\max}^{j}} \tag{4}$$

where

$$c_j(x) = \begin{cases} \max(0, g_j(x)) & j=1,...,m \\ \max(0, h_j(x) - \delta) & j=1,...,p \end{cases} \tag{5a}$$

$$c_{\max}^{j} = \max_{x} c_j(x) \tag{5b}$$

δ is the tolerance for handling equality constraints (usually 0.001). The constant m is the number of inequality constraint and p is for equality constraint. At least one or more constraint condition is not satisfied since every handled individual is infeasible. If the constraint violation $c_j(x)$ is smaller, the individual x does not violate the jth constraint and the corresponding effect on the total constraint violation $c_{\max}^{j}$ presents the same property. At the same time, $v(x)$ is divided by $m+p$, thus, the computation Eq. (4) can reflect the effect of the number of violation constraint for each individual.

The modified formulation of the i[th] objective function value for individual x is expressed as follows:

$$F_i(\mathrm{x}) = \overline{f}_{\max}^{i} + v(\mathrm{x}) \tag{6}$$

This modified formulation of objective presents some characteristics: I) Easy to implement with high efficiency; II) Flexible to deal with the objective function and constraint violation for each individual; III) The infeasible individual with large objective function value but low constraint violation is also given high priority.

4 Description of OSM

After FSM has explored vast search space and found sufficient feasible individuals, the subsequent operation is to conduct the OSM on the basis of the existing feasible individuals. In this study an adaptive particle swarm method for multi-objective optimization is adopted to execute this search mode.

4.1 PSEDO code of OSM process

Algorithm 3 shows the pseudo code of the OSM. For each feasible individual computes its speed using its local elitism and the global elitism (Line 1). Compute its new position based on its position at the last generation and speed (Line 2). The objectives and constraints of current feasible individual would be evaluated for the new position (Line 3). Lastly, the local elitism of this individual and the global elitism need to be updated because of the influence of the evaluation (Line 4).

Algorithm 3 The pseudo code of OSM

1. Compute the speed of one feasible individual using local elitism and global elitism;
2. Compute its new position;
3. Evaluate its objectives and constraints;
4. Update its local elitism and global elitism;

One precondition of the OSM is that all of the involved individuals must be always feasible, which means no constraints are violated and their decision variables are located at the feasible regions.

4.2 Description of improved elitist strategy

In order to fully take advantage of the non-dominated elitists to guide the swarm evolution, an exploration for the potential valuable information is required. This study concentrates on the stay history information for each elitist in the external archive. Therefore, an improved elitist archive is employed to store the elitists and their corresponding stay records. A

counter is accompanied with the elitist. This data structure is easy implementation and quite effective.

When a new elitist is produced it's appended in the elitist archive and its corresponding counter is assigned with 1. As the execution of the next iteration is over, the counter for each elitist in the archive increases by 1 if it's not dominated by the new particle, otherwise this elitist will be discarded from the archive. The stay history information for each elitist is traced by the counter, which represents how many iterations one elitist have stayed in the archive.

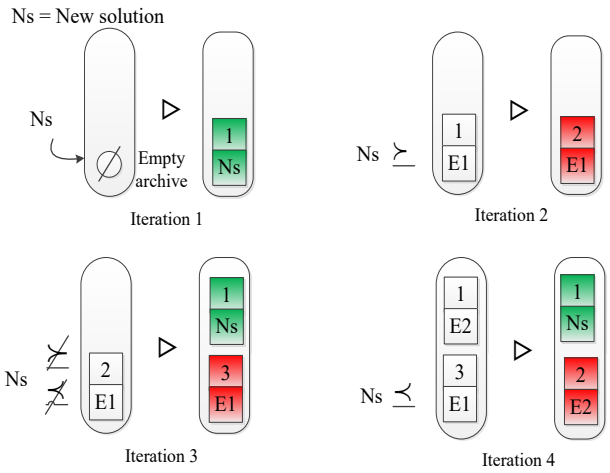

Fig. 1. Example of the change process of the improved elitist.

Figure 1 shows the changing process of the improved elitist archive while iterating. Ns represents a new solution, suppose only one non-dominated solution is produced at one iteration. At the first iteration, Ns is directly put into the empty elitist archive and its counter is tagged with 1. However, a new solution is refused since it's dominated by E1 at the second iteration and the counter of E1 becomes 2 at the same time. At the third iteration Ns and E1 are not dominated each other, therefore Ns is put into the archive as a new elitist and E1 is reserved with the value of 3 indicating E1 has stayed in the elitist for three iterations. At the fourth iteration there are already two elitists in the archive, E1 is discarded from the elitism archive because it's dominated by Ns, whereas E2 is reserved since they are not dominated each other. The life cycle of one elitist existing in the archive can be balanced by the counter.The relationship of E1 and E2 is elitist dominance. If E1 has a larger counter than E2, E1 has better performance in some aspect which is not exceeded until now.

Figure 2 shows an example of an elitist archive. At the first iteration it has four elitists and their counter values are 3, 2, 4 and 1 respectively. The total counter value is 10. For each elitist the selected probability value is the ratio of its counter value and the total counter value. Therefore, the selected probabilities of these elitists are 0.3, 0.2, 0.4 and 0.1 respectively. The elitist with larger counter would be selected with a bigger probability through this mechanism.

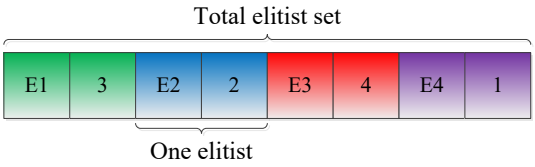

Fig. 2. An example of elitist archive.

Eq. (7) shows the formulas of the updates of speed and position. Where C_1 is the cognitive learning factor and represents the attraction that a particle has towards its own success; C_2 is the social learning factor and represents the attraction that a particle has towards the success of the entire population; W is the inertia weight, which is employed to control the impact of the previous history of velocities on the current velocity for each particle. $\vec{x}_{le_i}$ is the random local best position of the individual i in its local elitism; $\vec{x}_{ge_i}$ is one of the position with the oldest history stay in the global elitism; r_1 and r_2 are random values whose ranges are [0, 1].

$$\begin{aligned} \vec{v}_i(t+1) &= W\vec{v}_i(t+1) + C_1 r_1(\vec{x}_{le_i} - \vec{x}_i(t)) + C_2 r_2(\vec{x}_{ge_i} - \vec{x}_i(t)) \\ \vec{x}_i(t+1) &= \vec{x}_i(t) + \vec{v}_i(t+1) \end{aligned} \tag{7}$$

5 Experimental demonstration

5.1 Test problem and performance measure

In this study, the proposed algorithm is compared with NSGA-II on three classical constrained multi-objective problems and one real engineering case.

Srinivas and Deb used the following function with two variables and constraints [1]:

$$SRN: \begin{cases} \text{Min } f_1(x) = 2 + (x_1 - 2)^2 + (x_2 - 2)^2 \\ \text{Min } f_2(x) = 9x_1 - (x_2 - 1)^2 \\ \text{s.t.} \quad c_1(x) = x_1^2 + x_2^2 \le 225 \\ \qquad c_1(x) = x_1 - 3x_2 + 10 \le 0 \\ \qquad -20 \le x_1 \le 20 \\ \qquad -20 \le x_2 \le 20 \end{cases} \tag{8}$$

Tanaka suggested the following two-variable problem [1]:

$$TNK: \begin{cases} \text{Min } f_1(x) = x_1 \\ \text{Min } f_2(x) = x_2 \\ \text{s.t. } c_1(x) = x_1^2 + x_2^2 - 1 - 0.1\cos(16\arctan(x_1/x_2)) \ge 0 \\ \qquad c_2(x) = (x_1 - 0.5)^2 + (x_2 - 0.5)^2 \le 0.5 \\ \qquad 0 \le x_1 \le \pi \\ \qquad 0 \le x_2 \le \pi \end{cases} \tag{9}$$

Osyczka and Kundu [1] used the following six variable test problem with six constraints.

$$OSY:\begin{cases}\text{Min}\ \ f_1(x)=-(25(x_1-2)^2+(x_2-2)^2+(x_3-2)^2\\ +(x_4-2)^2+(x_5-2)^2)\\ \text{Min}\ \ f_2(x)=x_1^2+x_2^2+x_3^2+x_4^2+x_5^2\\ \text{s.t.}\ \ c_1(x)=x_1+x_2-2\geq 0\\ \qquad c_2(x)=6\text{-}x_1-x_2\geq 0\\ \qquad c_3(x)=2\text{-}x_2+x_1\geq 0\\ \qquad c_4(x)=2\text{-}x_1+3x_2\geq 0\\ \qquad c_5(x)=4\text{-}(x_3\text{-}3)^2-x_4\geq 0\\ \qquad c_6(x)=(x_5\text{-}3)^2+x_6-4\geq 0\\ \qquad 0\leq x_1,x_2,x_6\leq 10, 1\leq x_3,x_5\leq 5, 0\leq x_4\leq 6\end{cases} \tag{10}$$

Moreover, a real instance for two-bar truss design is also used to be tested [16]. The truss must carry on a certain load without elastic failure as shown in Figure 3. Therefore, except for the objective of constructing the truss for minimum volume, there are extra objectives of minimizing stresses in AC and BC, which can be handled as constraints. The optimization model is constructed as follows.

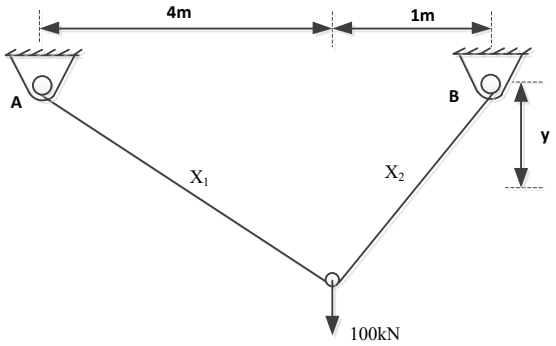

Fig. 3. The two bar truss.

$$TwoBarTruss:\begin{cases}\text{Min}\ \ f_1(x)=x_1\sqrt{16+y^2}+x_2\sqrt{1+y^2}\\ \text{Min}\ \ f_2(x)=\max(\sigma_{AC},\sigma_{BC})\\ \text{s.t.}\ \ \max(\sigma_{AC},\sigma_{BC})\leq 10^5\\ \qquad 1\leq y\leq 3\ \text{ and }\ x\geq 0\\ \qquad \sigma_{AC}=\dfrac{20\sqrt{16+y^2}}{yx_1}\\ \qquad \sigma_{BC}=\dfrac{80\sqrt{1+y^2}}{yx_2}\end{cases} \tag{11}$$

The additive epsilon indicator $I_{\varepsilon+}$ is applied to measure the algorithm performance, which is defined as following.

$$I_{\varepsilon+}(A,B)=\max_{a\in A}\ \min_{b\in B}\ \max_{1\leq i\leq k}(f_i(a)-f_i(b)) \tag{12}$$

In which A and B are two sets of approximation solutions to the Pareto optimal front. $I_{\varepsilon+}(A,B)$ means that the value of the objectives of the approximation solutions in B should be added, if the approximation solutions in A want to dominate the solutions in B. If $I_{\varepsilon+}(A,B)$ is smaller than $I_{\varepsilon+}(B,A)$, which means A is better than B.

5.2 Experimental results

Figure 4 - 7 show the optimal fronts using HSM and NSGAII for these four test problems respectively. It can be seen that HSM has better performance, including the number of solutions and the diversity of solution distribution. In particular, the algorithm performance with HSM is better than NSGA-II in test case of Osyczka2 and TwoBarTruss. These experiments demonstrate the effectiveness and priority of HSM.

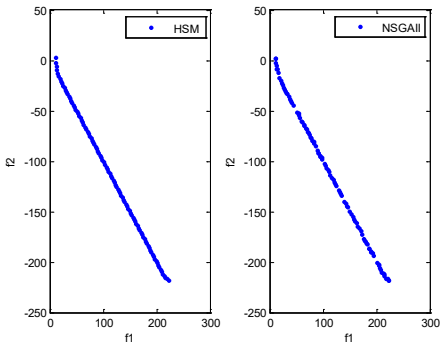

Fig. 4. The optimal fronts of Srinivas.

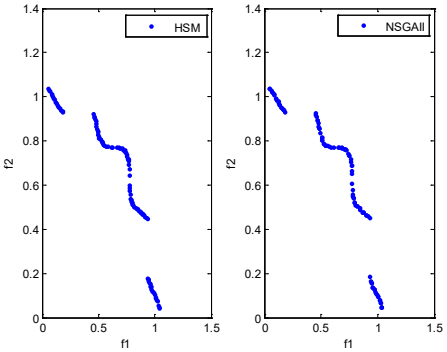

Fig. 5. The optimal fronts of Tanaka.

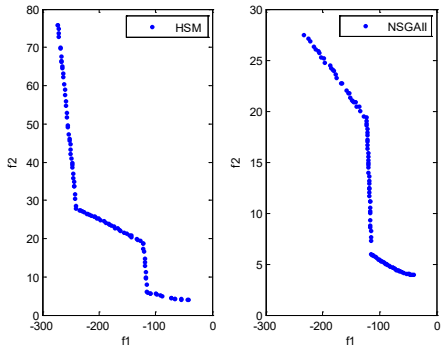

Fig. 6. The optimal fronts of Osyczka2.

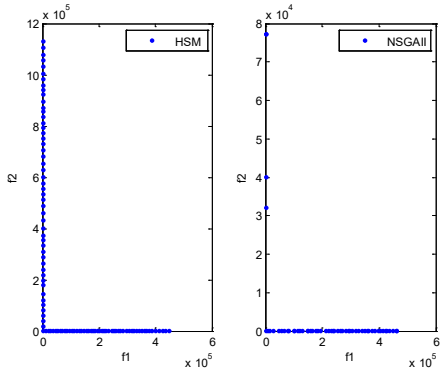

Fig. 7. The optimal fronts of TwoBarTruss.

In addition, the algorithm performance of HSM is evaluated by statistical method. Figure 8-11 show the boxplots of $I_{\varepsilon+}$ for these test problems respectively. It can be seen that HSM has smaller values than NSGAII for each metric, which means the front of HSM approximate to the true Pareto front with a better performance.

5.3 Influence analysis of two thresholds

Two threshold LOW_INF_S and $HIGH\_INF\_S$ are defined to distinguish different situations of population distribution, which represent the percentage of infeasible

individuals in the whole population. The thresholds determine different PDSs and have an imperative influence on the execution time for each search mode and the final results. In these experiments, the aforementioned four examples are tested, in which case of Srinivas is the loosest whereas case of Osyczka2 is the most demanding. Table 1 shows the statistical experimental results, in which the first column is the configuration value of LOW_INF_S and $HIGH\_INF\_S$ respectively. Each experiment is conducted in 50 runs.

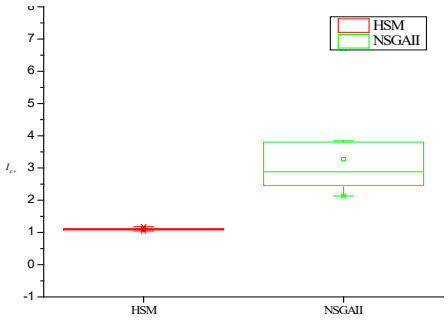

Fig. 8. The boxplot of $I_{\varepsilon+}$ for Srinivas.

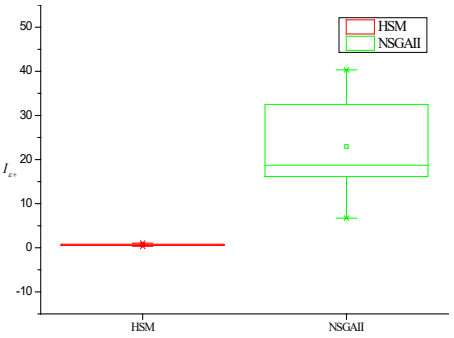

Fig. 9. The boxplot of $I_{\varepsilon+}$ for Osyczka2.

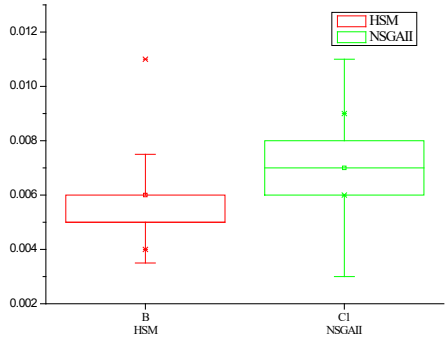

Fig. 10. The boxplot of $I_{\varepsilon+}$ for Tanaka.

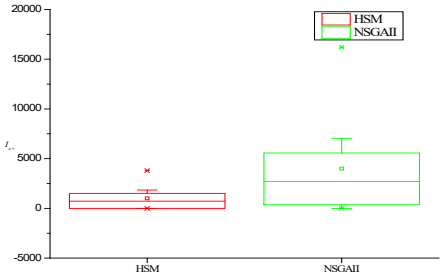

Fig. 11. The boxplot of $I_{\varepsilon+}$ for TwoBarTruss.

It can be summarized that: I) Case of Srinivas has the largest ratio whereas case of Osyczka2 is the smallest for most of test cases, which indicates that the test case with less constraints can obtain more feasible individuals; II) The ratio of the feasible individuals for all of the test cases reach over 80%, which indicates the proposed strategy is effective for all of the test problems; III) With the value of LOW_INF_S increasing the ratio value presents smaller value overall, this may because OSM is launched earlier with smaller value of LOW_INF_S. it can be roughly concluded that $HIGH\_INF\_S$ should be configured with a smaller value to ensure OSM can be executed successfully.

Table 1. Ratio of feasible individuals for different configurations of two thresholds.

	Srinivas	Osyczka2	Tanaka	Truss
(0.1,0.7)	95.3%	91.2%	91.4%	91.6%
(0.1,0.8)	95.1%	90.8%	91.2%	91.1%
(0.1,0.9)	94.5%	90.5%	90.9%	90.3%
(0.2,0.7)	92.6%	89.6%	90.85	90.5%
(0.2,0.8)	91.5%	88.65	89.5%	89.7%

(0.2,0.9)	90.6%	87.9%	89.4%	88.4%
(0.3,0.7)	91.2%	82.1%	85.4%	83.4%
(0.3,0.8)	90.1%	81.35	83.5%	82.45
(0.3,0.9)	89.4%	79.5%	81.25	81.5%

6 Conclusions

In this study a hybrid search mode for constrained multi-objective is proposed. The most obvious characteristic is that the processes of feasibility solution search and optimal solution search are separated completely. This strategy ensures the simplicity and uniqueness for each process. Moreover, the eventual individuals can be feasible also optimal with a theoretical proof. To improve this work in the future the strategies for each mode should explored further.

References

1. Deb K, Pratap A, Agarwal S and Meyarivan T, “A fast and elitist multiobjective genetic algorithms: NSGA-II,” J. IEEE Trans Evol Comput, vol. 40, no.8, pp.182-197,2002.
2. Du H Z, Xia N, Jiang J G et al, “A Monte Carlo enhanced PSO algorithm for optimal QoM in multi-channel wireless networks,” Journal of Computer Science and Technology, VOL.28, no.3,pp. 553-563, 2013.
3. Kennedy J and Eberhart R, “Particle swarm optimization,” In Proceedings os IEEE International Conference on Neural Networks, 1995, pp. 1942-1948.

4. Harada K, Sakuma J, Ono I, and Kobayashi S, “Constraint-handling method for multi-objective function optimization: Pareto descent repair operator,” In Proc. Int. Conf. Evol. Multi-Criterion Opt.,Matshushima, Japan, 2007, pp. 156–170.
5. Young N, “Blended ranking to cross infeasible regions in constrained multi-objective problems,” in Proc. Int. Conf. Comput. Intell. Modeling, Control and Automation, and Int. Conf. Intell.Agents, Web Technologies and Internet Commerce, Sydney, Australia. 2005. pp. 191–196.
6. Aidanpaa J, Anderson J, and Angantyr A.. “Constrained optimization based on a multi-objective evolutionary algorithm,” In Proc. Cong. Evol. Comput., Canberra, Australia. 2003. pp. 1560–1567.
7. Yuan W Q, Liu Y S, Zhao J J and Wang H W, “Pattern-based integration of system optimization in mechatronic system design,” Advances in Engineering Software, Vol.98, pp.23-37, 2016.
8. Dash R, Sa P K, Majhi B, “Particle swarm optimization based support vector regression for blind image restoration,” Journal of Computer Science and Technology, Vol.27, pp.989-995, 2012.
9. Mani A, “A novel hybrid constraint handling technique for evolutionary optimization,” IEEE Congress on Evolutionary Compuation, 2009.
10. Omeltschuk L, “Heterogeneous constraint handling for particle swarm optimization,” IEEE Congress on Evolutionary Compuation, 2012.
11. Zhang H B, Rangaiah G P, “An efficient constraint handling method with integrated differential evolution for numerical and engineering optimization,” Computers and Chemical Engineering, Vol.37, pp. 74-88, 2012.
12. Zhang C J, Lin Q, Gao L and Li X Y, “Backtracking search algorithm with three constraint handling methods for constrained optimization problems,” Expert Systems with Applications, Vol42, no.31, pp.7831-7845,2015.
13. Mario G F, Eduardo R T and Gregorio T P. “Constraint-handling through multi-objective optimization: The hydrophobic-polar model for protein structure prediction,” Computers & Operations Research, Vol.53, pp.128-153, 2015.
14. Liu LZ, Mu H B and Yang J H. “Generic constraints handling techniques in constrained multi-criteria optimization and its application,” European Journal of Operational Research, Vol.244,no.2, pp. 576-591, 2015.
15. Clerc, M, “The swarm and the queen: towards a deterministic and adaptive particle swarm optimization,” In: Proceedings of the IEEE congress on evolutionary computation, vol. 3, pp. 1951–1957, 1999.
16. P. Sabarinath, M.R. Thansekhar, R.Saravanan, “Multi Objective Design Optimization of Two Bar Truss Using NSGA-Ⅱ and TOPSIS,” Advanced Materials Research, Vol. 984, pp.419-424, 2014.