International Journal of Multimedia and Ubiquitous Engineering
Vol.10, No.5 (2015), pp.383-394
http://dx.doi.org/10.14257/ijmue.2015.10.5.36

An Improved Multi-objective Evolutionary Algorithm for Multi-Objective 0/1 Knapsack Problem

Zhanguo Li[1] and Qiming Wang[2]

1Software Engineering School of Pingdingshan University, Pingdingshan University, Pingdingshan, Henan province, China
2Computer Science and Technic Academy Department of Pingdingshan University, Pingdingshan University, Pingdingshan, Henan province, China
lizhanguo01@sohu.com

Abstract

To further enhance the distribution uniformity and extensiveness of the solution sets and to ensure effective convergence of the solution sets to the Pareto front, we proposed a MOEA approach based on a clustering mechanism. We named this approach improved multi-objective evolutionary algorithm (LMOEA). This algorithm uses a clustering technology to compute and maintain the distribution and diversity of the solution sets. A fuzzy C-means clustering algorithm is used for clustering individuals. Finally, the LMOEA is applied to solve the classical multi-objective knapsack problems. The algorithm performance was evaluated using convergence and diversity indicators. The proposed algorithm achieved significant improvements in terms of algorithm convergence and population diversity compared with the classical NSGA-II and the MOEA/D.

Keywords: *Multi-objective optimization; Multi-objective evolutionary algorithm; Knapsack problem; Clustering*

1. Introduction

A variety of engineering applications involve multi-objective optimization problems (MOPs). In addition, there are often mutual conflicts between objectives. Therefore, many real-world engineering optimization problems have multiple mutually conflicting objective functions [1]. Multi-objective optimization would compromise these multiple objectives. In such cases, a set of Pareto optimization solution sets can be derived. Due to the complexity of MOPs, traditional methods from operations research alone cannot produce solutions to these problems [2]. An evolutionary algorithm is a randomized optimization method that simulates the natural evolution process. Evolutionary algorithms have high degrees of parallel mechanisms that can optimize multiple objectives simultaneously. Thus, one run of an evolutionary algorithm can derive multiple Pareto optimal solutions. Moreover, evolutionary algorithms are not limited by the Pareto front shapes and continuities. This situation makes evolutionary algorithms the most suitable algorithms for solving MOPs. As a result, a new multi-objective evolutionary algorithm (MOEA) has been established [3]. MOEAs have been proven to be among the most effective methods for solving MOPs. The goal of multi-objective optimization is to let the solution set distribute evenly and quickly as it approaches a real Pareto front. Scholars have proposed a number of effective MOEAs based on this goal. MOEA research can be divided into three stages. In the first stage, Pareto domination is adopted to design a simple fitness function. The algorithms used in this stage include the multi-objective genetic algorithm (MOGA) proposed by Fonseca and Fleming [4], the niched Pareto genetic algorithm (NPGA) proposed by Horn and Nafpliotis [5], and the non-dominated sorting genetic algorithm (NSGA) proposed by Srinivas and Deb [6]. The second stage

ISSN: 1975-0080 IJMUE

uses elite groups to maintain the distribution from the Pareto optimal solution distribution. The algorithms used in this stage include the Pareto archiving evolutionary strategy (PAES) proposed by Knowles and Core [7], the strong Pareto evolutionary algorithm (SPEA) and the improved SPEA (SPEA2) proposed by Zitzler and Thiele *et al.* [8,9], and the improved NSGA (NSGA-II) proposed by Deb and Pratap *et al.* based on an elite strategy [10]. The third stage is characterized by a number of mixed algorithms generated via the combination of a MOEA and other novel intelligent algorithms. These mixed algorithms include the immune multi-objective optimization algorithm (IMOA) [11], the multi-objective particle swarm optimization algorithm (MOPSO)[12], and the quantum-inspired multi-objective evolutionary algorithm(QMEA) [13]. As stated in the previous studies [15-16]. To further enhance the distribution uniformity and extensiveness of the solution sets and to ensure effective convergence of the solution sets to the Pareto front, we proposed a MOEA approach based on a clustering mechanism.

2. Key Concepts

2.1 Multi-objective Optimization

Definition 2.1: A MOP is composed of *n* variable parameters, *K* objective functions, and *M* constraint conditions. The optimization objective is described by

$$\max\{y_1 = f_1(x), y_2 = f_2(x), ..., y_q = f_q(x)\}$$
$$s.t.\ g_i(x) \le 0, i = 1,2,...,m \quad (1)$$

where $x \in R^n$ is a vector with *n* decision variables, $f_i(x)$ is an objective function, and $g_i(x)$ are the m inequality constraint functions that form a feasible solution set.

Definition 2.2: The feasible solution set X_f is a set composed of decision vectors that satisfy all of the constraint conditions described as follows.

$$X_f = \{\vec{x} \in X \mid e(\vec{x}) \le 0) \quad (2)$$

Without loss of generality, we focus on the minimisation problem. For the maximisation problem, the definition is similar to the above definition.

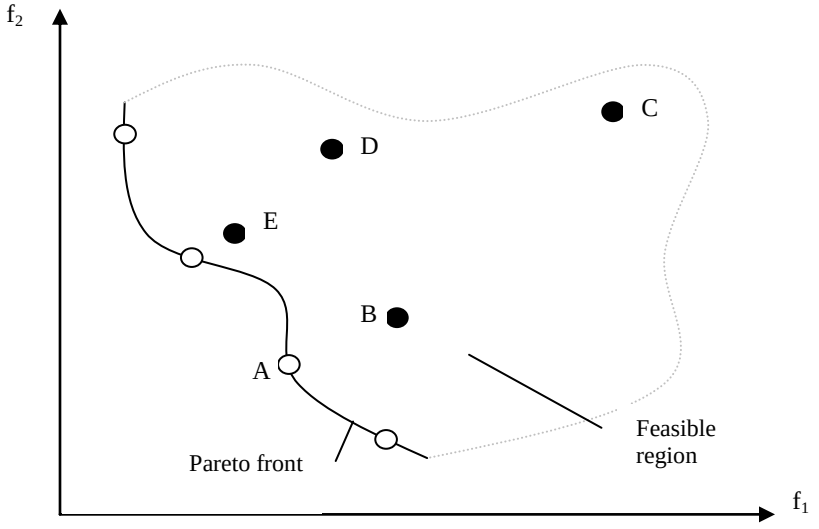


Figure 1. The Pareto Optimal Solution and the Pareto Front

In Figure 1, the solution represented by point E is worse than the solutions represented by Points C and D in terms of the two performance indicators. Based on the definition of Pareto dominance, solution E dominates solutions C and D. Thus, solution E is superior to

solutions C and D. By comparing solution E and B, it can be found that solution E is superior to the solutions C and D in terms of the performance indicator 1. By comparing solutions E and B, it can be found that solution E is superior to solution B in terms of performance indicator 1, but is worse than solution B in terms of performance indicator 2. Therefore, solution E is no better than solution B. Based on the above concept, the concepts involved in MOP can be described as follows.

Definition 2.3: Pareto dominance for decision vectors $\vec{x}, \vec{y}$:

(1) If and only if $\forall i \in \{1,2,...,K\}$, $f_i(\vec{x}) \le f_i(\vec{y})$ and $\exists j \in \{1,2,...K\}$, $f_i(\vec{x}) < f_i(\vec{y})$, then $\vec{x} \prec \vec{y}$ ($\vec{x}$ dominates $\vec{y}$).

(2) If and only if $\forall i \in \{1,2,...,K\}$ and $f_i(\vec{x}) \le f_i(\vec{y})$, then $\vec{x} \le \vec{y}$ ($\vec{x}$ weakly dominates $\vec{y}$).

(3) If and only if $\vec{x} \not\prec \vec{y} \wedge \vec{y} \not\prec \vec{x}$, then $\vec{x} \sim \vec{y}$ ($\vec{x}$ is no different from $\vec{y}$).

Definition 2.4: If and only if $\nexists \vec{a} \in A : \vec{a} \prec \vec{x}$ and $A \subseteq X_f$ is non-dominant, the decision vector $\vec{x} \in X_f$. If $\vec{x}$ is non-dominant in a feasible solution test X_f, then we call $\vec{x}$ the Pareto optimal solution.

Definition 2.5: Assuming the set $A \subseteq X_f$, $p(A)$ is the collection of non-dominant solutions in A according to the following expression.

$$p(A) = \{\vec{a} \in A \mid \nexists b \in A, b \prec a\} \quad (3)$$

In this case, $f(p(A))$ is called the A non-dominant front end and $p(A)$ is the non-dominant solution set for A.

2.2 Classical Literature Review

A. Genetic algorithms with sorting fitness assignment (MOGA)

MOGA is an evolutionary algorithm based on the Pareto optimization concept proposed by Fonseca and Fleming in 1993[4]. It determines an individual's advantages and disadvantages based on the dominating situation inside the whole population. MOGA also adopts a fitness assignment strategy based on sorting, such that the population can quickly converge to the Pareto front. To use the method, an individual's Pareto rank is first computed based on

(4)

where $rank(x_i, t)$ represents the rank of individual x_i for *t*-th generation of evolution and $p_i^{(t)}$ represents the number of individuals that are superior to x_i in the *t*-th generation population. When there is no other individual in the population dominating over the individual, then the rank for the individual is 1. Figure 2 shows an example using hierarchical ranking. An individual's rank is not necessarily continuous. For example, no individuals are associated with rank=4. The next step is to use proper functions to assign values to individuals with different ranks. The smaller the rank, the better the fit will be. If the same ranks correspond to multiple individuals, then the fitness share is needed to perform the fitness assignment. Goldberg noted that the MOGA fitness assignment method is a static fitness assignment strategy. Thus, it is easy to produce a larger selection pressure, which can lead to premature convergence. Therefore, MOGA adopts a sharing function and a niche technology to improve the population diversity. Through the Pareto optimal domain size and the population size, it is possible to determine the sharing radius and niche parameter σ_{share}. The solution vector sets with distances between objective

vectors smaller than σ_{share} will share fitness values. The main advantages of MOGA include its ease of execution and high efficiency.

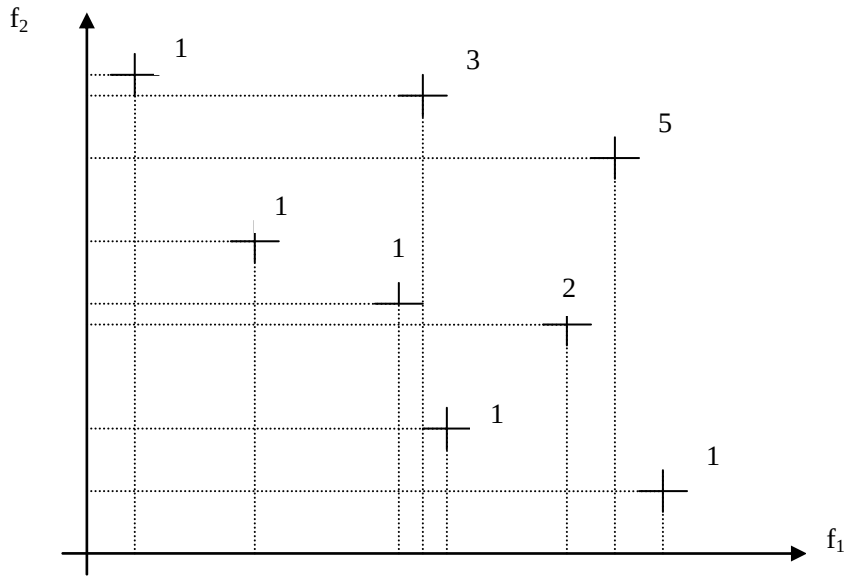


Figure 2: Sorting Individuals for the MOGA Algorithm according to Rank

B. Non-dominated sorting in genetic algorithms and improved algorithm (NSGA, NSGA-II)

NSGA was proposed by Srinivas and Deb in 1994[6]. It designed for multi-layer classification on a multi-objective solution population. Before the pairing selection for every generation, sorting can be performed based on the individual's Pareto relation. A sharing function method based on the decision vector space is then introduced to maintain the population diversity. Before the selection operation for the population, classification is performed based on an individual's Pareto and dominance conditions. All of the Pareto individuals are categorised as one class, and the same fitness is assigned to all Pareto solutions. Next, based on the sharing method proposed by Goldberg and Deb *et al.*, every Pareto-optimal solution niche number can be calculated based on formulas (5) and (6). Finally, for the individual, the original fitness is divided by the niche number to obtain the shared fitness. Therefore, for Pareto solutions located on the same Pareto front, both the niche numbers and the final sharing fitness values should be different, as expressed by

$$Sh(d_{ij}) = \begin{cases} 1-(\frac{d_{ij}}{\sigma_{share}})^2, if\ \ d_{ij} < \sigma_{share} \\ \\ 0, \qquad otherwise \end{cases} \tag{5},$$

where d_{ij} represents the distance between individual i and individual j, σ_{share} is the maximum allowed distance between individuals in the same niche, and $Sh(d_{ij})$ is the sharing function value when the distance is d_{ij} according to

$$m_i = \sum_{j \in Pop} Sh(d_{ij}) \tag{6},$$

where m_i is the niche number for individual i.

NSGA has three disadvantages: (1) high computational complexity, (2) use of a non-elite strategy, and 3) the need to set the sharing parameters. Based on the NSGA Pareto

sorting method, every individual should be compared with other individuals for every objective. Thus, the maximum number of comparisons required is MN^3, where *M* is the number of objectives and *N* is the population size. To reduce the computation complexity, Deb and Pratap *et al.* proposed an improved NSGA in 2000 named NSGA-II[10]. This algorithm adopts an elite strategy and requires the maximum number of comparisons of MN^2. In doing so, it greatly improves the computation speed. However, this algorithm has a higher demand on system storage performance.

In addition, NSGA-II uses a crowded comparison operator $\prec_n$ to solve the problem of sorting individuals with the same ranking during Pareto solution sorting, thus avoiding the need to set the sharing parameter σ_{share}. When several Pareto solutions with the same ranking exist, we used formula (7) to determine their partial order priority relations according to

$$i \prec_n j \quad if\ (i_{rank} < j_{rank}) \vee ((i_{rank} = j_{rank}) \wedge (i_{distance} < j_{distance})) \qquad (7),$$

where $rank$ is the Pareto sorting level and $distance$ is the crowding distance. Therefore, NSGA-II can overcome the following disadvantages found in NSGA and other algorithms: (1) the excessive computational load for a non-dominant classification process, (2) the lack of an elite-preservation mechanism, and (3) the difficulties in selecting the sharing parameter σ_{share}.

C. Pareto archived evolution strategy (PAES)

The objective of designing PAES was to provide a local search operation that adopts the same method to process all of the Pareto optimum solution points [7]. In PAES, a parent body and an offspring individual search solution space are adopted. This is a type of evolutionary strategy based on a (1+1) local search, which adopts a population size of 1 to construct a history set based on the previously found solution plan.

The algorithm initially generates an individual for the subsequent evaluation of all sub-objective functions. After the comparison of the new individual and a mutation with its parent individual, one of the Pareto individuals is selected. If two individuals have the same Pareto and cannot be compared, the new individual can be compared with the existing archived solution plan. It is worth mentioning that the uniqueness of PAES is its network crowding mechanism, *e.g.*, the commonly referred to exclusion process, that maintains the population diversity. This type of crowding operator is different from the usual crowding and niche computation methods. The exclusion process divides the objective space using an iterative method. Every solution is placed in a grid based on its objective value. Once the Pareto solution is identified, it is ready to be assigned to an archived set. If the Pareto solution point network number is relatively low, it will replace the individual associated with the maximum grid number in the archived geometry. The complexity of the PAES algorithm is defined by $O(\alpha MN)$, where α is the archive length, M is the number of objectives, and N is the population size. PAES is an important MOEA that has very good evolutionary performance and convergence speed. Many MOEAs described later in this paper are based on PAES. Thus, the contribution of Knowles is indelible.

D. The strength Pareto evolutionary algorithm and improved algorithm (SPEA, SPEA2)

The SPEA algorithm saves the Pareto solutions found from the evolution process in an external set and introduces the concept of concentration to perform a fitness assignment on individuals in the external Pareto solution set and the current population [8]. Assuming

that the current population is P , the external Pareto solution set is $P^{/}$ for any $i \in P^{/}, j \in P$ according to

$$f_i = si \quad (8)$$

and

$$f_j = 1 + \sum si \quad (9),$$

where $si = \frac{n}{N+1}$ is the concentration, n is the number of individuals P worse than individual i , and N is the total number of individuals in P . Obviously, $f_i \in [0,1), f_j \in [1, N)$. For SPEA, individuals associated with a smaller fitness have a higher chance of selection. Using this assignment strategy, a larger number of individuals in a niche will result in a higher Pareto solution concentration related to the niche. In addition, the results for the individuals inside the niche will have a higher fitness. Therefore, SPEA does not need to configure the distance parameter to achieve the goal of fitness level sharing. Considering that an exceedingly large external Pareto set $P^{/}$ will reduce the selection pressure, search speed, and local search, SPEA adopts an average coupled cluster method to control the size of $P^{/}$.

SPEA2 offers the following three improvements [9]:

(1) in terms of the individual fitness assignment strategy, not only does it consider the condition of how the current individual is better than the other individuals, but it also considers the condition where the current individual is worse than the other individuals;

(2) it adopts the nearest neighbouring individual density to evaluate technologies for improving the search accuracy;

(3) it proposes a new external Pareto solution updating algorithm to ensure that the boundary solutions are preserved.

SPEA2 redefines the concentration based on

$$s_i = |\{ j \mid j \in P_t + \overline{P} \wedge i \succ j \}| \quad (10),$$

where i and j represents any individual in the current population P_t and external Pareto solution set $\overline{P_t}$ and s_i is the concentration for individual i .

Formula (11) defines individual i's original fitness level according to the following expression.

$$r_i = \sum s_j \quad (11)$$

To differentiate individuals with the same original fitness level, Formula (12) defines the individual density as

$$d_i = \frac{1}{\sigma_i^k + 2} \quad (12),$$

where σ_i^k denotes the sum of distances from *k* individuals that are the closest to individual *i* and k usually takes on the value of $\sqrt{N + \overline{N}}$, where N and $\overline{N}$ are the sizes of the current population and external Pareto solution set, respectively.

Thus, individual *i*'s fitness is described by the following equation.

$$f_i = r_i + d_i \quad (13)$$

The main idea of the SPEA2 external Pareto solution set updating algorithm is that assuming the size of external Pareto solution set is $\overline{N}$, after every evolution of the current population and the original Pareto solution set, $\overline{N}$ number of individuals with the smallest fitness are selected to replace the individuals in the original Pareto solution set.

3. An Improved Multi-objective Evolutionary Algorithm Framework

3.1 Clustering Individuals in the Population to Form Solution Clusters

Assuming the population individual set $X = \{x_1, x_2, ..., x_n\} \subset R^s$ is a finite individual set among n individuals in the population individual space, $x_j = (x_{j1}, x_{j2}, ..., x_{jd}) \in R^s$ is individual x_j 's characteristic vector or pattern vector and x_{jk} is the value assignment for the *k*-th characteristic of the characteristic vector x_j . Cluster analysis on a given individual set X produces X 's C partition. Using the most classical cluster method based on the FCM partition algorithm, the optimization objective is represented by

$$J_M(U,V) = \sum_{i=1}^{C}\sum_{j=1}^{N} u_{ij}^m d_{ij}^2 \quad (14),$$

where $(d_{ij})^2 = \left\| x_j - v_i \right\|^2 = (x_j - v_i)^T (x_j - v_i)$ denotes the Euclidean distance for individual x_j and the cluster centre v_i . In $V = \{v_i\}(1 \le i \le c)$, v_i is the cluster centre and m is the weighted index ($m > 1$). The FCM partition matrix is the individual's membership matrix, which usually can be denoted as $U = [u_{ij}] \in R^{cn}$ (fuzzy partition matrix), where u_{ij} $(1 \le i \le c, 1 \le j \le n)$ denotes the membership of the j -th individual x_j belonging to *i*-th category, which satisfies the following constraint condition.

$$\sum_{i=1}^{C} u_{ij} = 1 \ , \ 1 \le j \le n$$

$$0 \le u_{ij} \le 1 \ , \ 1 \le i \le c \ , \ 1 \le j \le n \quad (15)$$

The FCM algorithm framework is shown in **Table 1.**

Table 1. The Fuzzy C-means Clustering Algorithm

Input: First, the number of clusters C is given based on $2 \le C < n$, where n is the number of individuals in the population. Next, the cluster centre **V**(0) is randomly initialised with the number of iterations $t = 0$.

Repeat:

1. $t = t + 1$;

2. Based on formula $u_{ij} = 1 \Big/ \sum_{k=1}^{c} \left(\frac{d_{ij}}{d_{kj}} \right)^{\frac{2}{m-1}}$, compute partition matrix **U** ;

3. Based on formula $v_i = \sum_{j=1}^{n} u_{ij}^m x_j \Big/ \sum_{j=1}^{n} u_{ij}^m$, compute cluster centre matrix **V** ;

4. Compare $v_i^{(t)}$ and $v_i^{(t+1)}$; if $\left\| v_i^{(t)} - v_i^{(t+1)} \right\| \le \varepsilon$, then stop iteration and output the partition matrix **U** and cluster the centre matrix **V** .

Otherwise Go back to Step 1.

3.2 Overall Algorithm Framework

The LMOEA algorithm framework is shown in Table 2.

Table 2: the improved multi-objective evolutionary algorithm（LMOEA）

Step1: Population initialization

Step2: Choose an initial individual x uniformly from $X = \{0,1\}^n$, $P \leftarrow \{x\}$

Step3: Calculate the fitness value of each individual for K fitness

Repeat:

1. Clustering individuals in the population to form solution clusters by the fuzzy C-means clustering algorithm**;**

2. Create offspring $x^{'}$ by crossover and mutation;

4. After several local improvement in the cluster, the the intra-cluster individuals are evaluated by evaluation functions of the individual;

5. Calculate the fitness value of each individual for K fitness

6. Non-dominated sorting operation

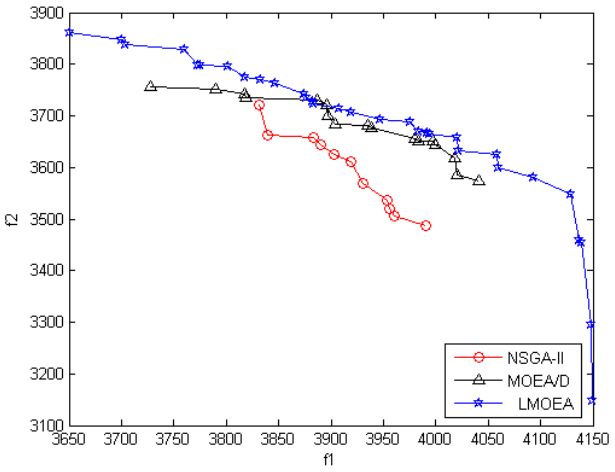
7. if not $\exists z \in P$ such that $(z \succ x^{'} \vee f(x^{'}) = f(z))$ then $P \leftarrow P \cup \{x^{'}\}$

Until meet the termination conditions, otherwise go back repeat.

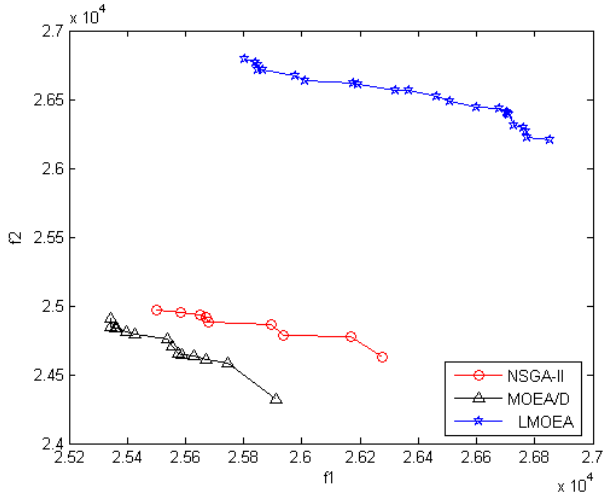
4. Experimental Results and Analysis

To verify the effectiveness and feasibility of the algorithm proposed in this paper, the authoritative data obtained from Zitler's webpage (http://www.tik.ee.ethz.ch/~zitzler/testdata.html) was used. The hardware platform for the experiment was a Pentium (R) (CPU 1.0 GHz, 1G RAM). The programming software was Matlab 7.0. The selected numbers of knapsacks were 2 and 3, and the numbers of items were,100, 250, 750. For each of the knapsack problems in the experiment, the LMOEA proposed in this paper, NSGA-II [10], and MOEA/D [17] were applied independently. The values for large and small S values contained in the space after running every algorithm were recorded. Figure 3(a) shows the non-dominant solution distribution results derived from the three algorithms after running 100 generations for an experiment with 2 knapsacks and 100 items. Figure 3(b) shows the non-dominant solution distribution

results from the three algorithms after running 100 generations with 2 knapsacks and 750 items.



(a).（2, 100）



(b).（2, 750）

Figure 3. Pareto Optimal Solution Distribution (100 Generation)

From Figure 3, it can be seen that compared with NSGA-II and MOEA/D, the search space determined by the LMOEA proposed in the paper is broader and the solution quality is higher. This result was due to the learning development approach introduced into the LMOEA, which offered an effective combination of global and local searches. The global search is deductive for the evolution moving towards the Pareto front, while the local search can be used to explore more feasible solutions in unknown regions. Therefore, the combination of these two can effectively maintain solution diversity, making the algorithm converge quickly and effectively. From Figure 5, it can be vividly seen that the solutions derived using the algorithm proposed in this paper are significantly superior to those produced by NSGA-II and MOEA/D in terms of convergence and uniformity.

Table 3. S of the Three Algorithms (100 Statistics)

	S(NSGA-II)	*S*(MOEA/D)	*S*(LMOEA)
（2,100）	4.278e06	4.897e06	5.561e06

（2,250）	7.011e07	7.599e07	8.673e07
（2,750）	6.100e08	6.378e08	7.385e08
（3,100）	3.981e10	4.345e10	4.601e10
（3,250）	6.448e11	6.887e11	6.901e11
（3,750）	1.785e13	1.776e13	1.901e13

Table 3 shows that after 100 independent runs for the knapsack problems, the averages of the LMOEA-derived S values are all greater than those derived from NSGA-II and MOEA/D. This result indicates that compared with NSGA-II and MOEA/D, the Pareto optimal solution set derived by the LMOEA has higher diversity and a more uniform distribution. Furthermore, the above experimental results demonstrate that the Pareto curve derived by the LMOEA displays a more uniform distribution and a higher convergence. Thus, the solution accuracy of the LMOEA is higher.

5. Conclusion

In this study, we explore a new MOEA approach. The proposed algorithm achieved significant improvements in terms of algorithm convergence and population diversity compared with the classical NSGA-II and the MOEA/D. The expansion of the application scope and the theoretical analysis of the algorithm for solving optimization problems are of great significance, providing avenues for future research.

References

[1] C A. Coello Coello, "Evolutionary multi-objective optimization: a historical view of the field". *IEEE Computational Intelligence Magazine*, vol. 1, (**2006**), pp. 28-36.
[2] A. Zhou, B.-Y. Qu and H. Li, "Multi-objective evolutionary algorithms: a survey of the state of the art", *Swarm and Evolutionary Computation*, vol. 1, (**2011**), pp. 32-49.
[3] J. D. Schaffer, "Some experiments in machine learning using vector evaluated genetic algorithms", *Ph.D. thesis, Vanderbilt University*, Nashville, TN, (**1984**).
[4] C. M. Fonseca and P. J. Fleming, "Genetic algorithms for multi-objective optimization: formulation, discussion and generalization". //*In: Proceedings of the 5th International Congress on Genetic Algorithms*. San Mateo, (**1993),** pp. 416-423.
[5] J. Horn and N. Nafpliotis, "A niched Pareto genetic algorithm for multi-objective optimization". *//In: Proceedings of International Congress on Evolutionary Computation*, Piscataway, (**1994**), pp. 40-45.
[6] N. Srinivas and K. Deb, "Multi-objective optimization using non-dominated sorting in genetic algorithms", *Evolutionary Computation*, vol. 2, (**1994),** pp. 221-248.
[7] J. D. Knowles and D. W. Corne, "The Pareto archived evolution strategy: a new baseline algorithm for multi-objective optimization", *//In: Proceedings of International Congress on Evolutionary Computation*. Washington, (**1998**), pp. 98-105.
[8] E. Zitzler and L. Thiele, "Multi-objective evolutionary algorithms: a comparative case study and the strength Pareto approach", *IEEE Transaction on Evolutionary Computation*, vol. 3, (**1999**), pp. 257–271.
[9] E. Zitzler, M. Laumanns and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm", *//Technische Informatik und Kommunikationsnetze (TIK)*, (**2001**).
[10] K. Deb, S. Agrawal and A. Pratap, "A fast and elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II", *IEEE Transaction on Evolutionary Computation*, vol. 6, (**1992**), pp. 182-197.
[11] I. Aydin, M. Karakose and E. Akin, "A multi-objective artificial immune algorithm for parameter optimization in support vector machine", *Applied Soft Computing*, vol. 11, (**2011**), pp. 120-129.
[12] C. A. Coello Coello and M. S. Lechuga, "MOPSO: a proposal for multiple objective particle swarm optimization", *// In: Proceedings of the 2002 Congress on Evolutionary Computation (CEC'02)*, (**2012**), pp. 1051-1056.
[13] K.-H. Han and J.-H. Kim, "Quantum-inspired evolutionary algorithm for a class of combinatorial optimization", *IEEE Transactions on Evolutionary Computation*, vol. 6, (**2002**), pp. 580–593.
[14] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization", *IEEE Transactions on Evolutionary Computation*, vol. 1, (**1997**), pp. 67–82.
[15] K. Deb, "Multi-objective optimization".//*Search methodologies. Springer US*, (**2014**), pp. 403-449.
[16] K. Sindhya, K. Miettinen and K. Deb, "A hybrid framework for evolutionary multi-objective optimization", *IEEE Transactions on Evolutionary Computation*, vol. 17, (**2013**), pp. 495-511.

[17] Q. F. Zhang and H. Li, "MOEA/D: A multi-objective evolutionary algorithm based on decomposition", *IEEE Transactions on Evolutionary Computation*, vol. 11, (**2007),** pp. 712-731.
[18] H. Li and Q. F. Zhang, "Multi-objective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II ", *IEEE Transaction on Evolutionary Computation*, vol. 13, (**2009),** pp. 284-302.

Authors

Zhanguo Li, he was born in shangshui County,Henan province,China in 1961.He received his Bachelor's degree in geophysical prospecting from Guilin University of Technology, China in 1984. Now, he is working as a adjunct professor in Software Engineering School of Pingdingshan University, Pingdingshan, Henan province, China. His major fields of study include network engineering and network convergence.

Qiming Wang, Now, he is working as a lecturer in Software Engineering School of Pingdingshan University, Pingdingshan, Henan province, China. His major fields of study include computer image processing and pattern recognition.