Available online at www.ijpe-online.com

International Journal of Performability Engineering

vol. 14, no. 5, May 2018, pp. 891-898
DOI: 10.23940/ijpe.18.05.p7.891898

Improved NSGA-II for the Job-Shop Multi-Objective Scheduling Problem

Xiaoyun Jiang* and Yi Li

School of Economics and Management, Xiamen University of Technology, Xiamen, 361024, China

Abstract

Job-shop scheduling is essential to advanced manufacturing and modern management. In light of the difficulty of obtaining the optimal solution using simple genetic algorithms in the process of solving multi-objective job-shop scheduling problems, with maximum customer satisfaction and minimum makespan in mind, we constructed a multi-objective job-shop scheduling model with factory capacity constraints and propose an improved NSGA-II algorithm. This algorithm not only uses an improved elitism strategy to dynamically update the elite solution set, but also enhances the Pareto sorting algorithm to make density computations more accurate, thereby ensuring population diversity. An example is given to verify that this algorithm can effectively enhance global search capabilities, save computing resources, and lead to a better optimal solution. Using this algorithm for job-shop scheduling optimization oriented towards multi-objective decision-making can provide corporate executives with a scientific quantitative basis for management and decision-making, thereby enhancing their companies' competitiveness.

Keywords: job-shop scheduling; multi-objective optimization; NSGA-II

(Submitted on February 1, 2018; Revised on March 16, 2018; Accepted on April 27, 2018)

1. Introduction

Job-Shop scheduling is a major topic in the research of manufacturing systems. Its fundamental task is to find out how to achieve comprehensive optimization in one or multiple aspects of a decomposable job, such as product processing time and cost, by issuing production instructions, deploying resources for the operations, and setting processing time and processing sequence, while satisfying the constraint conditions, such as delivery time, resource availability and process routes, to the largest extent possible [10,17,23]. Among the various factors, job-shop scheduling is one of the greatest concerns as its goal is to optimize the allocation of the company's resources and improve its profitability. Normally, the soundness of a scheduling scheme is evaluated with indicators such as completion time, delivery delay, inventory, and cost [19,21]. In recent years, with the widespread adoption of the JIT notion, companies have become more concerned about how to complete their production processing tasks at a minimum cost within a specified timeframe, i.e., how to achieve multi-objective optimization with both delivery time and cost taken into consideration. This will be one of the most important indicators of the quality of a scheduling scheme.

Because of its typicality among production planning issues, job-shop scheduling has received much scholarly attention, leading to the emergency of a host of new theories and methods across various fields, such as management, computer science, machinery, automation, and mathematics. By the number of objectives, optimization methods are classified as single-objective or multi-objective. In scientific research and practical production, there are usually multiple optimization objectives. These objectives constrain one another, and the rise of one optimization objective often leads to the decline of at least another optimization objective. While taking all optimization objectives into consideration, decision-makers need to concentrate on one or more objectives according to the changing research background or production background. Hence, it is of great practical significance to develop an efficient multi-objective optimization method with a set of non-dominant optimal solutions.

* Corresponding author.
E-mail address: jxycom@163.com

Multi-Objective optimization algorithms are one of the fast-growing disciplines in the past 30 years. They have evolved from traditional multi-objective optimization methods, like the weighted method, the constraint method, and the objective programming method, to more classic multi-objective optimization methods developed by subsequent scholars on the basis of evolutionary phenomena in nature, such as the Multi-Objective Genetic Algorithm (MOGA) [2,3,4,5], the Vector Evaluated Genetic Algorithm (VEGA) [16,18], the Niched Pareto Genetic Algorithm (NPGA) [6], the Non-dominated Sorting Genetic Algorithm (NSGA) [8,13], the Particle Swarm Optimization Algorithm (PSO) [9,11,20], the Artificial Immune System (AIS) [12], and the NSGA-II [1,7,14]. In recent years, NSGA-II has attracted more attention for its high efficiency in providing multi-objective solutions. The single biggest advantage of NSGA-II over NSGA is the adoption of a greater elitism strategy based on the various non-inferior front ends generated, thus reducing the overall computational time of the algorithm [24]; NSGA-II also uses density estimation and density comparison operators in lieu of the original sharing mechanism to ensure the diversity of population and the distribution uniformity of pareto solutions [15,22], which can effectively solve the problem of multi-objective programming in various fields. However, NSGA-II has some shortcomings, such as low convergence accuracy, uneven Pareto front distribution, and slow convergence. How to speed up the solving process and obtain a satisfactory solution is a crucial matter for companies seeking to improve their competitiveness and images. Hence, it is imperative to improve NSGA-II by expanding its applicability, increase its convergence speed, and improve its solving accuracy. In this paper, we present an improved non-dominated sorting genetic algorithm II (INSGA-II) with significantly enhanced solving performance and expanded applicability.

The remainder of this paper is structured as follows: the mathematical model of Job-shop scheduling optimization is described in the second section. The INSGA-II is then presented in the third section. An illustrative example is used to demonstrate and evaluate the application and effectiveness of the proposed approach, which is described in the fourth section. Finally, some conclusion remarks are made in the fifth section.

2. Mathematical Model of Job-shop Scheduling Optimization

A job-shop scheduling optimization problem can be described as follows: A factory has multiple production lines that can make a variety of products. Any of the products can be processed on any of these lines. For the same product, the processing time varies from line to line due to differences in processing performance. After completing the production of a product, a line would need setup as if it was to produce another product, and the setup time varies from product to product. As there are different delivery periods for different clients, the factory must, within the constraints of its capacity, schedule production on the lines in a matter that would allow it to achieve the minimum load of machines (i.e., the minimum completion time) and the maximum client satisfaction (i.e., the shortest total delay of orders). For the sake of clarity, it is assumed that, for the orders of the same client, production can take place only on the same production line, and that the orders of each client require only product from the factory.

Suppose the factory has K production lines and has received L orders from clients. Assume that, for order *i*, the delivery period is t_i (i=1, 2, …, L), the processing time on the *k*-th production line is q_{ik} (i=1,2,…,L; k=1,2,…,K), and the setup time between order *i* and order *j* is d_{ij}, with the initial setup time assumed to be d_{0j} (i, j=1, 2, …, L). Let n_k be the number of orders (n_k = 0 denotes the unused *k*-th line) for processing on the *k*-th line; set R_k represents the *k*-th line, where element r_{ki} represents that the sequential number of order r_{ki} on the *k*-th line is *i*; let r_{k0} =0, denoting the initial status in which production has yet to begin. C_{ik} stands for the delay period of the *i*-th product on the *k*-th line. Based on the above assumptions, the following mathematical model is created for scheduling optimization:

$$
\begin{aligned}
&\min\Big(\sum_{k=1}^{K}\Big(\sum_{i=1}^{n_k} q_{r_{ki}k}\Big)\Big)\\
&\min\left[\sum_{k=1}^{K}\Big(\sum_{i=1}^{n_k} C_{ik}\Big)\right]\\
&s.t.\ \ C_{ik} = \max\Big(0, \sum_{j=1}^{i}\big(d_{r_{k(j-1)}r_{kj}} + q_{r_{kj}k}\big) - t_i\Big)\\
&0 \le n_k \le L \qquad \forall k = 1,...K\\
&\sum_{k=1}^{K} n_k = L\\
&R_k = \big\{r_{ki} \mid r_{ki} \in \{1,2,...,L\}, i = 1,2,...,n_k\big\}\\
&R_{k_1} \cap R_{k_2} = \varnothing \ \ (\forall k_1 \ne k_2)
\end{aligned}
\tag{1}
$$

3. Optimization Algorithm

3.1. Algorithm Step

We propose an INSGA-II, which is described below, to solve the scheduling problem. The main steps of the proposed INSGA-II can be summarized as follows. The simplified flowchart is displayed in Figure 1.

- Step 1. Set control parameters, such as the population size N, the crossover mode, the crossover fraction C, the mutation rate M, and the maximum evolutionary number of generations GN, according to the data size and data distribution characteristics.
- Step 2. Initialize the population. For the initial population with a random production scale of N, set the initial elite solution set E as an empty set and the initial generation index to 0 (see 3.3 for details).
- Step 3. Execute the crossover procedure according to the crossover fraction C to obtain new population P' (see 3.4 for details).
- Step 4. Execute the mutation procedure according to the mutation rate M to obtain new population P''.
- Step 5. Calculate the fitness of the population after P'' and E are merged (see 3.6 for details), and then execute the Pareto sorting algorithm (see 3.7 and 3.8 for details) to obtain subset population Q (see 3.9 for details).
- Step 6. Update elite solution set E (see 3.9 for details).
- Step 7. g=g+1. If the maximum number of generations is reached, select the result in E for output; otherwise, go to Step 3.

3.2. Coding Method

The chromosomes are coded in integers, and each value of the chromosomes represents the order number. When there are L orders to be processed and K production lines, individual chromosomes are represented as a matrix of K rows and L columns. For example, for a factory with 9 orders to be completed on 3 production lines, a production scheme can be constructed using a matrix of 3 rows and 9 columns randomly arranged from 1 to 9 and 0, as shown by chromosome $\begin{bmatrix} 5 & 0 & 0 & 3 & 0 & 0 & 6 & 0 & 0 \\ 0 & 2 & 0 & 0 & 4 & 0 & 0 & 0 & 9 \\ 0 & 0 & 1 & 0 & 0 & 8 & 0 & 7 & 0 \end{bmatrix}$. Note that each row contains only one none-zero element.

3.3. Initial Population

The sequence from 1 to L is randomly generated, and each bit is randomly assigned to the K row. The remaining positions are filled with 0, forming a chromosome; N different chromosomes thus form the initial population P (the population size is set to be N).

3.4. Crossover Procedure

For crossover, two chromosomes are selected from the population by the crossover fraction C randomly, and then crossover positions are randomly selected for crossover by row. Following crossover, a certain order number of the chromosomes is duplicated, while another order number is missing, making it necessary to change the duplicated order number to the missing corresponding order number.

3.5. Mutation Procedure

Moderate mutation, which is to maintain the diversity of the individuals in the population, can improve the speed of the algorithm. For mutation, mutant individuals are first selected from the population by the mutation rate M randomly, and then mutation positions are randomly selected to swap positions with the chromosomes by row.

3.6. Fitness Evaluation

To determine the fitness of the corresponding production scheme of a chromosome, it is necessary not only to find out whether it satisfies the constraint condition of production, but also to calculate its two objective function values, as show in Eq. (1). The coding method used in this paper can ensure that all orders will be put into production and satisfy the constraint condition that the orders of the same client can only be produced on the same production line. Then, the objective function value is calculated, and for each individual, two fitness values are obtained – the total load of the machine and the total delay period of the order.

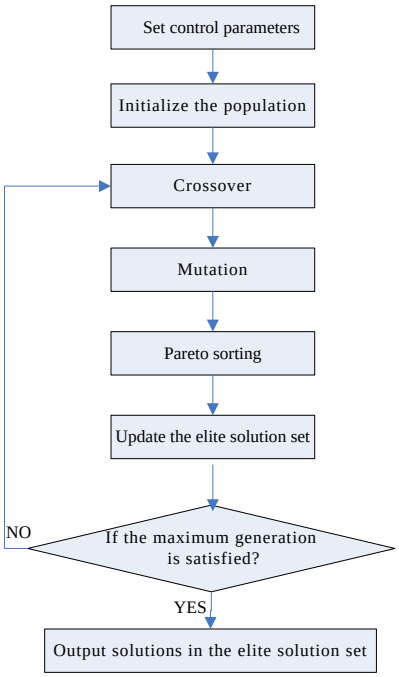

Figure 1. The flow chart for the proposed approach

3.7. Mutation Procedure

For each chromosome x_i in the population, first set two parameters b_i and S_i, with b_i being the number of the solutions of dominating chromosome x_i in the population, and S_i being the set of solutions dominated by x_i. The specific Pareto sorting algorithm is as follows:

- Step 1. Find the individuals in the population whose $b_i = 0$ and deposit them into the current non-dominated solution set W.
- Step 2. For each individual x_j in the current non-dominated solution set W, run through all the dominated individual set S_j, and subtract each individual k in set S_j by 1, i.e., subtract the number of individuals whose solutions dominate individual x_k by 1 (because individual x_j, which dominates individual x_k, has been deposited into the current non-dominated solution set W); if $b_k - 1 = 0$, deposit individual x_k into another set V.
- Step 3. Make W a set of first-rank non-dominated individuals; hence, the individual solutions in W are optimal; it dominates the individuals and is not dominated by any other individuals. Assign the same non-dominated sequential order 1 to all individuals in the set and executive the above ranking procedure for set V. Assign the corresponding non-dominated sequential order 2 until all individuals have been ranked, i.e., all individuals have been assigned a corresponding non-dominated sequential order.

3.8. Density Computations

As density computations are crucial to ensuring the diversity of the population, it's necessary to calculate the density of the set of non-dominated individuals at each rank. The steps for density computations are as follows:

- Step 1. Perform domination ranking for the population according to the optimization objectives, and let the density of the two individuals on the frontier to be infinitely large.
- Step 2. Calculate the density of other individuals in the population using Eq. (2). Suppose x_1 and x_2 are two individuals in a non-dominated solution set V of a certain rank. Density d_i can be calculated using Eq. (2).

$$d_i = \sqrt{\sum_{j=1}^{Z} \left(\left| f_j^{i+1} - f_j^{i-1} \right| \right)} \tag{2}$$

where Z is the number of optimization objectives (2 for this paper); f_j^{i+1} means the value of *j*-th objective function at the *i*+1 point; and f_j^{i-1} refers to the value of the *j*-th objective function at the *i*-1 point.

3.9. Elitism Strategy

The present algorithm includes an elitism strategy, which not only prevents the loss of outstanding individuals during the evolution of the population, but also accelerates the convergence of the algorithm. In the algorithm proposed in this study, an elite solution set is preserved in the evolution process at all times; it is inserted into the current population and participates in the evolution process of crossover and mutation. To save resources, the maximum number of solutions in the solution set is $\lfloor N/4 \rfloor$, where N is the population size. The steps of updating elite solution E for each generation are as follows:

- Step 1. Perform the following steps for each solution x_i in sub-generation population Q. If x_i is dominated by any solution in E, try the next solution x_{i+1}. Otherwise, remove all the solutions that are dominated by x_i from E and then put x_i into E.
- Step 2. If the number of solutions in E exceeds $\lfloor N/4 \rfloor$, calculate density using the method in 3.8, sort the solutions in order of density, and remove excess solutions; the number of solutions is $\lfloor N/4 \rfloor$.

3.10. Summary

The INSGA-II method presented in this paper has the following advantages over NSGA-II:

- INSGA-II includes an improved elitism strategy to dynamically update the elite solution set in each generation of evolution and to maintain a fixed number of solutions to save computing resources.
- INSGA-II incorporates an improved Pareto sorting algorithm, which improves the accuracy of density computations and enables the individuals of the population in the quasi-Pareto domain to spread evenly over the entire Pareto domain, thus ensuring the diversity of the population.

4. Experiment and Computations

On the basis of the above algorithm, we develop a program using Matlab 2016a and perform computations with an experiment for the following real example.

A certain factory uses three production lines to manufacture six products and now it needs to schedule production for 30 orders from clients. Suppose that the products and delivery period required for each order are as given in Table 1; the setup times are as given in Table 2, with the first line being the initial setup time and the setup time for the same product being 0; and the processing time on different production lines for each product is as given in Table 3. In the experiment, we use the following parameter settings. The values of the population size N and the maximum number of generations GN are both 100, and the values of the crossover fraction and the mutation rate are 0.9 and 0.1 respectively.

Table 1. Ordered Products and Delivery Periods

Order No.	Product No.	Delivery Period (days)	Order No.	Product No.	Delivery Period (days)	Order No.	Product No.	Delivery Period (days)
1	5	34	11	1	51	21	5	29
2	5	33	12	4	50	22	6	59
3	1	60	13	4	13	23	1	41
4	4	2	14	3	13	24	1	36
5	2	36	15	6	30	25	6	57
6	5	36	16	5	8	26	6	4
7	5	29	17	6	5	27	1	7
8	1	56	18	4	51	28	6	29
9	1	52	19	6	42	29	4	14
10	3	15	20	5	33	30	5	21

Table 2. Setup Time for the Products (days)

Setup Time / Product No. (Product No.)	1	2	3	4	5	6
0	1	3	1	3	3	1
1	0	3	3	3	3	3
2	3	0	3	1	1	3
3	3	3	0	3	3	3
4	3	1	3	0	3	3
5	3	3	3	3	0	3
6	1	1	3	3	3	0

Table 3. Processing Times for Products on Different Lines (days)

Product Line / Product No.	1	2	3	4	5	6
1	3	1	5	1	5	3
2	4	4	2	3	3	4
3	1	2	2	4	1	3

Table 4. Scheduling Configuration Schemes for 30 Products

Scheme	Scheduling Configuration	Total Delay Period (days)	Machine Total Load (days)
A	Product Line 1: 17, 26, 28,13,19, 24, 25,18, 22,12 Product Line 2: 4, 29,15, 21,9,3 Product Line 3: 27, 2, 30,16,14,10,1, 7, 6, 20,8,5,11, 23	8	62
B	Product Line 1: 17, 26,13, 29,19, 25,18,22, 24,9,12 Product Line 2: 4,15, 28 Product Line 3: 27, 2, 30,16, 21,14,10,1, 7,8,6, 20,5,11,3, 23	9	55
C	Product Line 1: 17, 26,13, 29,18, 22, 9, 25,12 Product Line 2: 4,15,19, 28 Product Line 3: 27, 2, 30,16, 21,14,10,1, 7, 24,6, 20,5,8,11,3, 23	10	54
D	Product Line 1: 17, 26,13, 4, 29,19, 25, 28,9,18,12, 22 Product Line 2: 8,15 Product Line 3: 27, 30,16, 21,14,10,1, 7, 24,6, 20, 2,5,11,3, 23	13	53
E	Product Line 1: 17, 26,13, 4, 29, 25,12,18,9, 22 Product Line 2: 19,15, 28 Product Line 3: 27, 2, 30,16, 21,14,10,1, 7, 6, 20,5, 24,11,8,3, 23	15	52
F	Product Line 1: 17, 26,13, 4, 29,9, 25,18,19,5, 22,12 Product Line 2: 15, 28 Product Line 3: 27, 2, 30,16, 7,14,10,1, 21,8,6, 20, 24,11,3, 23	16	50

Using the algorithm program developed by Matlab, we obtain a Pareto solution set that consists of six scheduling configuration schemes for 30 products, as shown in Table 4. Their distribution in the target space is shown in Figure 1, with the transverse coordinate and the vertical coordinate standing for the total delay time and the total load of the machine, respectively. From Figure 1, we can see that the optimal Pareto solution found by INSGA-II forms the Pareto frontier, indicated by the red dots, with other dots being the solutions whose non-dominated sequence is 1 as obtained from the Pareto sorting in each generation of search. In Figure 2, the total delay time of optimization result A is the highest among all results, but its total machine load is not the lowest; on the contrary, the total machine load of optimization result F is the lowest among all results, but the total delay time is not the lowest. All optimization results have the best total delay and total machine load, which is the characteristic of Pareto solution sets. Managers can choose a result that satisfies their needs according to their companies' actual production situation. If the company stresses client satisfaction and seeks to reduce total delay time while not being particularly concerned about the average load of machines, optimization result A can be picked; otherwise, optimization result F can be chosen. Optimization results B...E are tradeoff options.

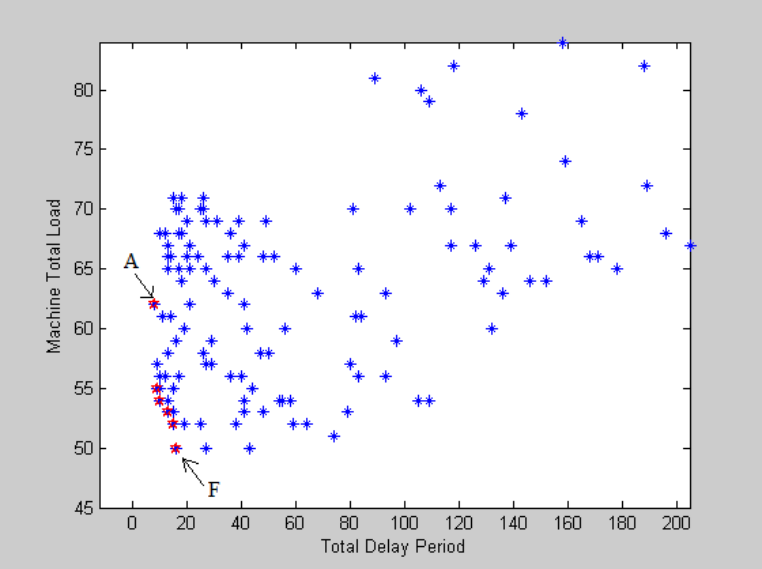

Figure 2. Distribution of Pareto Solutions in the Objective Space

5. Conclusions

This paper presents an improved NSGA-II algorithm. This particular algorithm not only incorporates an improved elitism strategy to dynamically update the elite solution set and thus save computing resources, but also improves the Pareto sorting algorithm to enhance the accuracy of density computations and thus, ensure the diversity of the population. When applied to research on multi-objective job-shop scheduling problems, this algorithm features rapid convergence and high solution quality in solving large-scale complicated scheduling problems. The application of this method provides corporate executives with a sound quantitative basis for management and decision-making and helps them solve job-shop scheduling problems; moreover, it enables companies to achieve optimization for key objectives, such as productivity, client satisfaction, and production cost, thereby enhancing their competitiveness.

Acknowledgements

This work was financially supported through grants from Longyan Science and Technology Bureau (No. 2015LY29). The authors thank the 3 anonymous reviewers for their helpful suggestions.

References

1. S. Chatterjee, K. Abhishek and S. S. Mahapatra, "NSGA-II Approach of Optimization to Study the Effects of Drilling Parameters in AISI-304 Stainless Steel," *Procedia Engineering*, vol. 97, pp. 78-84, 2014
2. K. Deb, "Multi-Objective Optimization Using Evolutionary Algorithms," John Wiley & Sons, New York, 2001
3. K. Deb, A. Pratap and S. Agarwal, "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182-197, June 2002
4. C. M. Fonseca and P. J. Fleming, "Genetic Algorithms f or Multiobjective Optimization: Formulation Discussion and Generalization," in *Proceedings of the International Conference on Genetic Algorithms*, pp. 416-423, California, US, June 1993
5. J. Gao, M. Gen, L. Sun and X. Zhao, "A Hybrid of Genetic Algorithm and Bottleneck Shifting for Multi-Objective Flexible Job Shop Scheduling Problems," *Computers and Industrial Engineering*, vol. 53, no. 1, pp. 149-162, 2007
6. J. Horn, N. Nafpliotis and D. E. Goldberg, "Multiobjective Optimization Using the Niched Pareto Genetic Algorithm," in *Proceedings of the International Conference on Evolutionary Computation*, pp. 82-87, Piscataway, US, June 1994
7. R. F. Huang, X.W. Luo and B. Ji, "Multi-Objective Optimization of a Mixed-flow Pump Impeller Using Modified NSGA-II Algorithm," *Science China Technological Sciences*, vol. 58, no. 12, pp. 2122-2130, November 2015
8. X. Y. Jiang and W. C. Chen, "An Effective Approach for the Optimisation of Cutting Parameters", *International Journal of Computer Applications in Technology*, Vol. 50, No. 3/4, pp. 180–185, December 2014
9. X. Y. Jiang and H. H. Wu, "Optimization of Setup Frequency for TOC Supply Chain Replenishment System with Capacity Constraints", *Neural Computing and Applications*, vol. 23, no. 6, pp.1831-1838, November 2013
10. V. Kaplanoğlu, "An Object-Oriented Approach for Multi-Objective Flexible Job-Shop Scheduling Problem," *Expert Systems with Applications*, vol. 45, pp.71-84, November 2016
11. J. Kennedy, "Encyclopedia of Machine Learning: Particle Swarm Optimization," Springer US, New York, 2011
12. A. Kurapati and S. Azarm, "Immune Network Simulation with Multiobjective Genetic Algorithms for Multidisciplinary Design Optimization," *Engineering Optimization*, vol. 33, no. 2, pp. 245-260, March 2000

13. M. Y. Liu and E. B. Chen, "An Improved Differential Evolution Algorithm for Vehicle Routing Problem with Simultaneous Pickups and Deliveries and Time Windows", *Engineering Applications of Artificial Intelligence*, vol. 23, no. 2, pp. 188-195, 2010
14. A. Martínez-Vargas, J. Domínguez-Guerrero and A. G. Andrade, "Application of NSGA-II Algorithm to the Spectrum Assignment Problem in Spectrum Sharing Networks," *Applied Soft Computing*, vol. 39, pp. 188-198, May 2016
15. A. X. Ni, Y. F. Zhang and H. X. Chen, "An Improvement to NSGA-II Algorithm and Its Application in Optimization Design of Multi-element Airfoil," Acta Aerodynamica Sinica, vol.32, no.2, pp. 252-257, June 2014
16. R. Pérez-Rodríguez, S. Jöns and A. Hernández-Aguirre, "Simulation Optimization for a Flexible Jobshop Scheduling Problem Using an Estimation of Distribution Algorithm," *the International Journal of Advanced Manufacturing Technology*, vol. 73, no.4, pp.3-21, September 2014
17. M. Rohaninejad, A. Kheirkhah A and P. Fattahi, "A Hybrid Multi-Objective Genetic Algorithm Based on the ELECTRE Method for a Capacitated Flexible Job Shop Scheduling Problem," *the International Journal of Advanced Manufacturing Technology*, vol. 77, no. 4, pp. 51-66, September 2015
18. M. Sakawa and R. Kubota, "Fuzzy Programming for Multi-object Ive Job Shop Scheduling with Fuzzy Processing Time and Fuzzy Due Date through Genetic Algorithm", *European Journal of Operational Research*, vol. 12, no. 2, pp. 393-407, June 2000
19. J. D. Schaffer, "Multiple Objective Optimization with Vector Evaluated Genetic Algorithms," in *Proceedings of the International Conference on Genetic Algorithms and their Applications*, pp. 93-100, Pittsburgh, US, May 1985
20. X. Shao, W. Liu and Q. Liu, "Hybrid Discrete Particle Swarm Optimization for Multi-Objective Flexible Job-Shop Scheduling Problem," *the International Journal of Advanced Manufacturing Technology*, vol. 67, no. 9, pp. 2885-2901, 2013
21. N. Srinivas and K. Deb, "Muiltiobjective Optimization Using Nondominated Sorting in Genetic Algorithms," *Evolutionary computation*, vol.2, no.3, pp. 221-248, August 1994
22. L. Wang, G. Zhou and Y. Xu, "An Enhanced Pareto-Based Artificial Bee Colony Algorithm for the Multi-Objective Flexible Job-Shop Scheduling," *the International Journal of Advanced Manufacturing Technology*, vol. 60, no. 9, pp. 1111-1123, 2012
23. H. H. Wu and X.Y. Jiang, "Improved Genetic Algorithms for Optimization of Inventory Allocation in LED Chip Manufacturing Plants," *Journal of Interdisciplinary Mathematics*, vol. 20, no. 3, pp. 727-738, August 2017
24. L. K. Zhang, "A Two-Stage Optimization Algorithm of NSGA2 Guided by the Improved Bee Evolutionary Genetic Algorithm and Its Application", Hunan university, Hunan, 2016