International Journal of Multimedia and Ubiquitous Engineering
Vol.9, No.4 (2014), pp.317-326
http://dx.doi.org/10.14257/ijmue.2014.9.4.33

The Similar Algorithm Research of Concurrent Open-shop Scheduling

YanMin Ma

[1]*School of computer and information engineering, Harbin University of Commerce, Harbin, China*

mym654321@163.com

Abstract

The sparse linear method can reduce or expanse the solution range of integer programming problem by using former enumeration method lists all solution space, and then constraining them into a scale, after that picking up optimal solution form this scale. Calculation time will be reduced by using a revised integer programming model based on sparse linear, which can help reduce the number of solution space.

Keywords: *concurrent open shop scheduling; Sparse linear; NP*

1. Introduction

Concurrent Open-Shop Scheduling Problem is an important research branch of Open-Shop Scheduling. It has wide applications in many aspects, such as modern transport and logistics industry, modern service industry, large-scale systematic maintenance industry, clothing industry, health care and so on [1]. In recent years, artificial intelligence [2], computational intelligence, real-time intelligence and other research results have been applied to the solving process of Concurrent Open-Shop Scheduling Problem, which achieved remarkable results. Therefore, analysis of the Concurrent Open-Shop Scheduling Problem not only for plays an important role in promoting scheduling theory, but also has practical significance.

Differed from Flow Shop and Job, workpiece can be processed on machines by various order, no fixed processing order, so the number of its feasible solution is huge. Compared with the traditional open workshop, they have similar features. Their process is not bound by the orders, the difference between them is concurrent open-shop can make multiple machines work on a workpiece simultaneously, which has more feasible solution space, therefore most of the concurrent open-shop scheduling problem is NP-complete problem that makes us cannot get accurate optimal solution in time.

TEOFILO and SARTAJ [3] are considered the first two to put forward the definition of the open-shop scheduling and the open-shop scheduling application examples, and give the open workshop the detailed description, and introduce the interruptible and non-interruptible open-shop scheduling problem which proves that the two machine open-shop scheduling problem is polynomial complexity problem. Later scholars have found that most of the open-shop scheduling problem is NP problem; it can't give exact solution of the problem. Sergey. V. evastianov and Gerhard j. oeginger [4] used a polynomial time approximation algorithm to the solving of open-shop scheduling problem with the minimum manufacturing period.

Project Supported: social science research project of Heilongjiang province（12D056）; natural science foundation of Heilongjiang province(F201243); 2012 higher education teaching reform project of Harbin University of Commerce (201242)

Then many scholars proposed concurrent workshop model of the open-shop scheduling based on the open-shop scheduling. Concurrent open-shop scheduling model in 1979 was the earliest, [5] Baker put forward m completely machine parallel machine environment of workshop, and if the m value is an arbitrary value, will this problem markers for PD, m parallel machines in this paper refers to m machine can processing that can be together do concurrent processing. Recently, Roemer [6] for concurrent open-shop scheduling problem has carried on the classification of the different target summary, the complexity of the problem are given for different target.

NP is also called non-deterministic polynomial problem. Non-deterministic algorithm, which consists of conjecture and verification, should be defined firstly to the definition of NP problem. When dealing a satisfiability problem, a guess will be made by non-deterministic algorithm firstly, and the initial value will be granted, after that, whether the value meets the function could be known. If the function requirement is fulfilled, then the guess of non-deterministic algorithm is correct. If the time complexity of non-deterministic polynomial is a polynomial, then this algorithm is called as non-deterministic polynomial problem. Otherwise, if the problem can only be solved by non-deterministic polynomial problem, this problem is called as NP problem.

Sparse linear is a popular NP problem solving method in recent years, and can make the large scale of integer programming problem sparse appropriately. After sparseness, the integer programming model can reduce the number of feasible solution, and get the order of approximately optimal solution. (Near optimal solution)The near optimal solution and similarity between near optimal solution and optimal solution could be obtained and approved by ordering questions, after analyzing the tendency of the sparse result.

2. The Sparse Linear of Integer Programming Model

The relationship between sparse linear model and unrevised integer programming model is followed:

Goal function of integer programming model and integer programming model based sparse linear are the same, but the constraint parameters of sparse linear model turn from integer parameters into similar constant parameters. The complexity of integer programming model is reduced by minimizing or maximizing the scale the constraint condition of sparse linear model. But, the calculation result will be waived from the actual optimal solution after altering the constraint parameters and condition. In this paper, the sparse linear aims to reduce the constraints of integer programming to optimal solution, time complexity of integer programming model, and time consuming.

3. Integer Programming Model based on Workpiece's Completion Time

Workpiece completion time variable integer programming model is based on the workpiece completed time nodes, then we establish relationship between relevant workpiece completion time and processing time, the relationship between the different constraint conditions should be established by the unequal relationship between the workpiece completing time node and the beginning time of processing, such as workpiece-j should be processed after the workpiece-k, establishing the constraint relations with completion time variables can be expressed as: $c_{ik} \geq \max(s_{ik}, c_{ij}) + p_{ik}$

Inequalities are given work-piece k's completion time to greater than or equal to j's completion time which is ahead of k, and the work-piece ready time in the larger value of k, and sum of the processing time of k.

There is no accurate solution method for solving multi-objective problems now, thus for validating model, we can only model and solve to the target of the minimized manufacturing duration. The integer programming model according to the completion time variables is set up as follows, notes for IP3-1:

$$\text{minimize max } c_{ij} \forall j \in (1,2,\ldots,n), \forall i \in (1,2,\ldots,m) \qquad \textbf{(3-1)}$$

$$c_{ij} \geq c_{i'j} + p_{i'j} \text{ or } c_{i'j} \geq c_{ij} + p_{ij} \quad \forall j \in (1..n), \forall i \in (1..m), i, i' \in (1..m) \qquad \textbf{(3-2)}$$

$$c_{ij} \geq c_{ik} + p_{ij} \text{ or } c_{ik} \geq c_{ij} + p_{ik} \quad \forall j,k \in (1..n),\ j \neq k, \forall i \in (1..m) \qquad \textbf{(3-3)}$$

$$s_{ij} \geq r_j \quad \forall j \in (1..n), \forall i \in (1..m) \qquad \textbf{(3-4)}$$

$$c_{ij} = s_{ij} + p_{ij} \quad \forall j \in (1..n), \forall i \in (1..m) \qquad \textbf{(3-5)}$$

$$c_{ij} \text{ is integer} \quad \forall j \in (1..n), \forall i \in (1..m) \qquad \textbf{(3-6)}$$

$$s_{ij} \text{ is integer} \quad \forall j \in (1..n), \forall i \in (1..m) \qquad \textbf{(3-7)}$$

4. Sparse Linear of IP3-1

Sparse linear method, which is called Queyranne-Multi-Dimension Machine Scheduling Method of No Interruption, is used to integer programming model IP3-1 built to obtain the sparse linear constraints of one machine scheduling problem, which also means that the sparse linear based on formula (3-3). Although the constraints focus on one machine scheduling, it can also be used to Concurrent Open-Shop Model in this paper, and to get near optimal solution combined with algorithm based on principles.

To one machine scheduling problem, if no leisure exits in machine scheduled, every workpieces, which the total quantity of the workpiece is J, have to processed in that machine, the time of processing workpiece on each machine is $p_j\,(j \in J)$, the processing finished time of workpiece is c_j. Q is the feasible solution set in scheduling, q is one of solutions in Q, the processing order is q_1, $q_2,\ldots,q_n$, then the total quantity of feasible solution in Q is n!. According to an order of q, the $c_j = p_j$ or $c_j = p_j + p_{j-1}$ is reasonable, due to the no leisure time and no interruption processing.

$$\sum_{j=1}^{J} p_j c_j = p_1 \times p_1 + p_2 \times (p_1 + p_2) + p_3 \times (p_1 + p_2 + p_3) + \ldots + p_n \times (p_1 + p_2 + \ldots + p_n) \qquad \textbf{(4-2)}$$

The right side of the formula(4-1) simplifies to:

$$\frac{1}{2}\left(p_1^2+p_2^2+...+p_n^2\right)+\left\{\frac{1}{2}\left(p_1^2+p_2^2+...+p_n^2\right)+\left[p_1\times p_2+p_3\left(p_1+p_2\right)\right.\right.$$
$$\left.\left.+...+p_n\left(p_1+p_2+...+p_n\right)\right]\right\}=\frac{1}{2}\sum_{i-1}^{n}p_i^2+\left\{\frac{1}{2}p_1^2+p_1\times p_2+\frac{1}{2}p_2^2+p_3^2+2p_3\left(p_1+p_2\right)+\right. \tag{4-2}$$
$$\left.\frac{1}{2}\left(p_4^2+p_5^2+...p_n^2\right)+2p_4\left(p_1+p_2+p_3\right)+...+2p_n\left(p_1+p_2+...+p_{n-1}\right)\right\}$$

Formula (4-2) on the right side of the second part application of mathematical induction to prove:

$$\frac{1}{2}p_1^2+p_1\times p_2+\frac{1}{2}p_2^2+p_3^2+2p_3\left(p_1+p_2\right)+2p_4\left(p_1+p_2+p_3\right)+$$
$$...+2p_n\left(p_1+p_2+...+p_{n-1}\right)+\frac{1}{2}\left(p_4^2+p_5^2+...p_n^2\right)=\frac{1}{2}\left(\sum_{i=1}^{n}p_i\right)^2 \tag{4-3}$$

According to formula (4-1), formula (4-2), formula (4-3) simplifies to type:

$$\sum_{j=1}^{J}p_jc_j=\frac{1}{2}\sum_{i=1}^{n}p_i^2+\frac{1}{2}\left(\sum_{i=1}^{n}p_i\right)^2 \tag{4-4}$$

In multi machines concurrent processing problem, presumption that every machine makes constant processing to every workpiece, but as the existence of machine interruption, there will be workpiece preparation requirement, which causes the waiting spare time to workpieces and machines. So, the separate limitation of concurrent open-shop scheduling problem made as below, the sparse revised to function (3-3) is following:

$$\sum_{j=1}^{J}p_jc_j\geq\frac{1}{2}\sum_{i=1}^{n}p_i^2+\frac{1}{2}\left(\sum_{i=1}^{n}p_i\right)^2 \qquad \forall i\in\{1,2,...,m\} \tag{4-5}$$

$$c_{ij}\ \text{is integer} \qquad \forall j\in(1,2,...,n),\forall i\in(1,2,...,m) \tag{4-6}$$

The IP4-1 is the new integer programming model which consists of formula (3-1), (3-2), (4-4), (3-5) and (4-6). IP4-1 is sparse linear integer programming model of IP3-1. The new model turns formula (3-3), which means a kind of mechanical separation constraint, into formula (4-9), which means a machine can only process one workpiece at a period of time. This resource constraint results in a lot of discrete constraints, which is the main reason of the huge solution space. The calculation time will increase, as the increasing of solution and time-complexity. Applying formula (4-9) can improve the time complexity obviously and reduce the number of constraints, the problem of IP3-1, which caused by much constraints and large calculation time resulting from huge solution space, can be basically solved by changing the NP problem into a formula of polynomial time complexity. IP4-1 is as following:

$$\text{minimize } \max c_{ij} \qquad \forall j \in (1...n), \forall i \in (1...m) \tag{4-7}$$

$$c_{ij} \geq c_{i'j} + p_{i'j} \text{ or } c_{i'j} \geq c_{ij} + p_{ij} \quad \forall j \in (1...n),\ \forall i \in (1...m),\ \ i' \in (1...m) 且 i \neq i' \tag{4-8}$$

$$\sum_{j=1}^{J} p_j c_j \geq \frac{1}{2}\sum_{i=1}^{n} p_i^2 + \frac{1}{2}\left(\sum_{i=1}^{n} p_i\right)^2 \qquad \forall i \in \{1,......,m\} \tag{4-9}$$

$$s_{ij} \geq r_j \qquad \forall j \in (1...n), \forall i \in (1...m) \tag{4-10}$$

$$c_{ij} = s_{ij} + p_{ij} \qquad \forall j \in (1...n), \forall i \in (1...m) \tag{4-11}$$

$$c_{ij} \text{ is integer} \qquad \forall j \in (1...n), \forall i \in (1...m) \tag{4-12}$$

5 Simulation Analysis of Sparse Planning

This section will use five-group cases to simulate the integer planning model IP3-1 and IP4-1 of concurrent open-workshop scheduling , data of cases takes randomly, the matrix scales of workpiece processing time window, respectively, are 4*3, 5*4, 6*4, 6*6, 7*5. According to the constraints of model, we write CPLEX program, and give the examples of processing-time's matrix, respectively, into a simulation, simulation results data unit is minutes.

Example one:

A given instance of 4 pieces and 3 machines is in example one, and given processing time matrix shows in Table 5.1, additional constraints that machine 3 and machine 2 cannot be concurrent processing, table data unit is in minutes.

Table 5.1. Example-processing-time's matrix

	machine1	machine 2	machine 3	ready time
workpiece 1	3	7	5	1
workpiece 2	1	4	7	2
workpiece 3	8	2	6	1
workpiece 4	4	9	4	0

Example two:

A given instance of 5 pieces and 4 machines is in example two, and given processing time matrix shows in Table 5.2, additional constraints that machine 3 and machine 2 cannot be concurrent processing, table data unit is in minutes.

Table 5.2. Example-processing-time's matrix

	machine1	machine2	machine3	machine4	ready
workpiece 1	3	7	5	4	1
workpiece 2	1	4	7	2	2
workpiece 3	8	2	6	5	4
workpiece 4	2	6	9	2	3
workpiece 5	4	6	3	9	2

Example three:

A given instance of 6 pieces and 4 machines is in example three, and given processing time matrix shows in Table 5.3, additional constraints that machine 3 and machine 2 cannot be concurrent processing, table data unit is in minutes.

Table 5.3. Example-processing-time's matrix

	machine1	machine2	machine3	machine4	ready
workpiece 1	3	7	5	2	1
workpiece 2	1	4	7	3	7
workpiece 3	8	2	6	4	4
workpiece 4	2	6	9	5	3
workpiece 5	1	2	4	6	5
workpiece 6	2	7	9	3	4

Example four:

A given instance of 6 pieces and 4 machines is in example four, and given processing time matrix shows in Table 5.4, additional constraints that machine 3 and machine 2 cannot be concurrent processing, table data unit is in minutes.

Table 5.4. Example-processing-time's matrix

	machine1	machine	machine3	machine4	machine5	machine6	ready
workpiece1	6	7	5	3	9	4	1
workpiece2	2	8	0	8	1	3	2
workpiece3	7	3	6	5	4	5	0
workpiece4	2	5	9	1	8	7	3
workpiece5	5	2	9	4	7	9	2
workpiece6	4	2	8	3	1	7	1

Example five:

A given instance of 7 pieces and 5 machines is in example five, and given processing time matrix shows in table 5.5 additional constraints that machine 3 and machine 2 cannot be concurrent processing, table data unit is in minutes.

Table 5.5 example-processing-time's matrix

	machine1	machine2	machine3	machine4	machine5	ready time
workpiece1	2	7	6	4	6	1
workpiece2	1	4	7	2	3	2
workpiece3	8	2	6	3	1	4
workpiece4	4	9	4	6	7	1
workpiece5	3	2	8	2	2	3
workpiece6	5	6	4	1	4	4
workpiece7	3	9	4	6	4	7

Simulation of IP3-1, the calculation time, number of variables, and number of constraints are as Table 5.6

Table 5.6 The Performance Parameters of IP3-1

	number of variables	calculation time (second)	number of constraints	calculation speed（second)
Example 1	24	0.05	156	17948.4
Example 2	40	5.91	325	22114.8
Example 3	48	23.07	462	22737.4
Example 4	60	105.6	574	16575
Example 5	70	49865.6	784	17723

Simulation of IP4-1, the calculation time, number of variables, and number of constraints are as Table 5.7

Table 5.7 The Performance Parameters of IP4-1

	number of variables	calculation time (second)	number of constraints	calculation speed（second)
Example 1	24	0.03	60	1826.9
Example 2	40	0.05	84	1321.6
Example 3	48	0.11	108	1217.9
Example 4	60	0.24	136	2438.9
Example 5	70	0.37	168	5507.4

From Table 5.7, the number of constraints of sparse linear method has reduced. Calculation time has reduced as the reducing number of sparse integer programming model, all five numerical examples cost no more than 1 second. So, from the simulation comparison above, the calculation time has reduced a lot, as the replacement formula (3-3) with formula (4-5) and formula (4-6) in IP4-1 model.

The begin time and end time of processing are shown as Table 5.8 and Table 5.9 respectively, the end time of the fifth numerical example is 26 minutes. From the sparse result, it can't be the solution of scheduling problem, as it can't meet constraints, which include no concurrent work in interrupted machines of concurrent open-shop and time window of workpiece machining time.

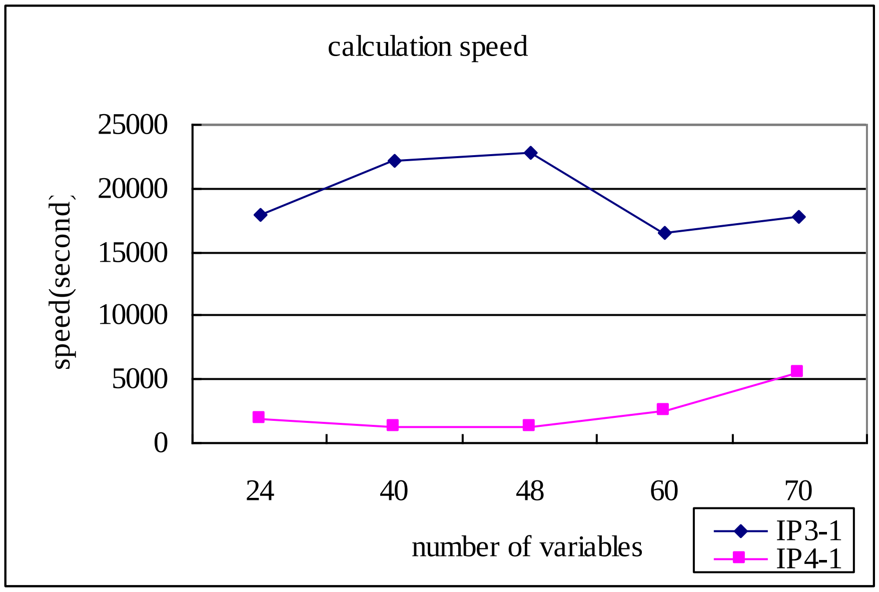


Figure 5.1 Calculation speed

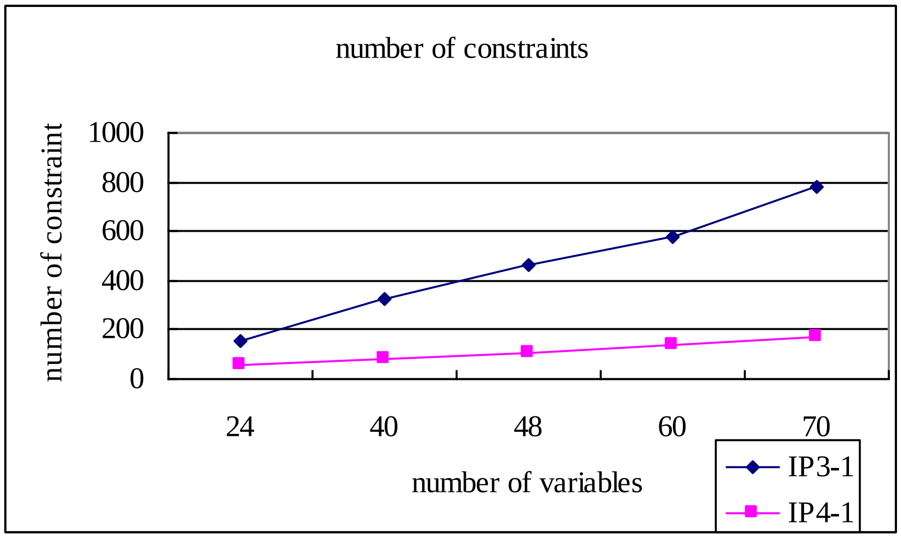


Figure 5.2 number of Constraints

Table 5.8 the Begin Time of Machining

	machine1	machine 2	machine 3	machine 4	machine 5
workpiece 1	18	11	20	15	6
workpiece 2	22	22	15	22	23
workpiece 3	4	17	19	4	21
workpiece 4	18	12	16	1	1
workpiece 5	21	23	12	23	24
workpiece 6	5	14	16	12	19
workpiece 7	7	13	14	7	7

Table 5.9. The End Time of Machining

	machine 1	machine 2	machine 3	machine 4	machine 5
workpiece 1	20	18	23	19	12
workpiece 2	26	26	22	24	26
workpiece 3	12	19	25	7	24
workpiece 4	22	21	18	7	8
workpiece 5	24	25	21	25	26
workpiece 6	10	20	20	16	16
workpiece 7	10	17	16	13	11

6. Conclusion

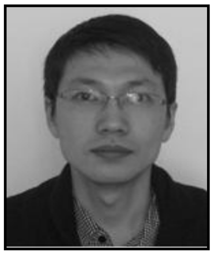
After making sparse linear to mathematics integer programming model of scheduling problem, the solutions can't meet the constraints of scheduling problem model, as the maximizing and minimizing of constraints of former model. These solutions can be the accordance of regular algorithm, but not the solutions to scheduling problem.

References

[1] J. Di, M.Yanmin, F. ZhiPeng and F. Xiaolin, "A RFID anti-collision algorithm based on multithread regressive-style binary system", 2012 International Conference on Measurement, Information and Control, **(2012)**, pp. 365-369.
[2] M. -Y. Chen, C. -N. Yang and C. -S. Laih, "Authorized Tracking and Tracing for RFID", IJSIA, vol. 1, no. 1, **(2007)** July, pp. 1-14.
[3] Q. Y. Dai, R. Y. Zhong, M. L. Wang, X. D. Liu and Q. Liu, "RFID-enable Real-time Multi-experiment Training Center Management System", IJAST, vol. 7, **(2009)** June, pp. 27-48.
[4] Y. M. Ma and D. Jin, "Anti-collision Algorithm based on Multi-threaded RFID Lock Position", International Journal of Hybrid Information Technology, vol. 6, no. 3, **(2013)**, pp. 95-104.
[5] B. King and X. Zhang, "Applying RFID to Secure the Pharmaceutical Supply Chain", IJSIA, vol. 1, no. 2, **(2007)** October, pp. 71-84.
[6] M. Modarres and M. Ghandehari, "Applying circular coloringto open shop scheduling", Scientia Iranica, vol. 15, no. 5, **(2008)**, pp. 652-660.
[7] S. G. Dastidar and R. Nagi, "Batch splitting in an assembly scheduling environment", Production Economics, vol. 105, no. 2, **(2007)**, pp. 372–384.
[8] R. L. Graham, E. L. Lawler, J. K. Lenstra and A. H. G. R. Kan, "Optimization and Approximation in Deterministic Sequencing and Scheduling: A Survey", Annals of Discrete Mathematics, vol. 5, **(1979)**, pp. 287–326.
[9] C. S. Sung and H. A. Kim, "A two-stage multiple-machine assembly scheduling problem for minimizing sum of completion times", Production Economics, vol. 113, **(2008)**, pp. 1038–1048.
[10] T. Gonzale and S. Sahni, "Open shop scheduling to minimize finish time", Journal of the Assooauon for Computing Machiner, vol. 23, no. 4, **(1976)**, pp. 665-679.
[11] T. A. Roemer and R. Ahmadi, "The Complexity of scheduling customer orders", INFORMS, Dallas, **(1997b)**.
[12] C. S. Sung and S. H. Yoon, "Minimizing total weighted completion time at a pre-assembly stage composed of two feeding machines", International Journal of Production Economics, vol. 54, no. 3, **(1998)**, pp. 247–255.
[13] C. T. Ng, T. C. E. Cheng and J. J. Yuan, "Concurrent open shop scheduling to minimize the weighted number of tardy jobs", Journal of Scheduling, vol. 6, no. 4, **(2003)**, pp. 405–412.
[14] H. L. Huang and B. M. T. Lin, "Concurrent openshop problem to minimize the weighted number of late jobs", Multiprocessor Scheduling, vol. 8, **(2007)**, pp. 215-220.
[15] T. Gonzale and S. Sahni, "Open shop scheduling to minimize finish time", Journal of the Assooauon for Computing Machiner, vol. 23, no. 4, **(1976)**, pp. 665-679
[16] M. Mastrolilli, M. Queyranne, A. S. Schulz, O. Svensson and N. A. Uhan, "Minimizing the sum of weighted completion times in a concurrent open shop", Operations Research Letters, vol. 38, no. 5, **(2003)**, pp. 405–412.
[17] G. Wang and T. C. E. Cheng, "Customer order scheduling to minimize total weighted completion time", Omega, vol. 35, no. 5, **(2005)**, pp. 409–416.

[18] M. Queyranne, “Structure of a simple scheduling polyhedron”, Mathematical Programming, vol. 58, **(1993)**, pp. 263-285.
[19] A. S. Schulz, “Scheduling to minimize total weighted completion time: Performance guarantees of LP-based heuristics and lower bounds”, Computer Science, vol. 1084, (1995), pp. 301-315.

Authors



YanMin Ma

YanMin Ma, master, lecturer, School of computer and information engineering, Harbin University of Commerce.