A New Approach in Job Shop Scheduling: Overlapping Operation

Parviz Fattahi[a,*], Mohammad Saidi Mehrabad[b]

[a] *Industrial engineering department, Faculty of engineering, Bu-Ali Sina University, Hamedan, Iran*

[b] *Industrial engineering department, Iran University of Science & Technology, Tehran, Iran*

Received 15 Dec., 2008; Revised 24 May., 2009; Accepted 15 Jun., 2009

Abstract

In this paper, a new approach to overlapping operations in job shop scheduling is presented. In many job shops, a customer demand can be met in more than one way for each job, where demand determines the quantity of each finished job ordered by a customer. In each job, embedded operations can be performed due to overlapping considerations in which each operation may be overlapped with the others because of its nature. The effects of the new approach on job shop scheduling problems are evaluated. Since the problem is well known as NP-Hard class, a simulated annealing algorithm is developed to solve large scale problems. Moreover, a mixed integer linear programming (MILP) method is applied to validate the proposed algorithm. The approach is tested on a set of random data to evaluate and study the behavior of the proposed algorithm. Computational experiments confirmed superiority of the proposed approach. To evaluate the effect of overlapping considerations on the job shop scheduling problem, the results of classical job shop scheduling with the new approach (job shop scheduling problem with overlapping operations) are compared. It is concluded that the proposed approach can improve the criteria and machines utilization measures in job shop scheduling. The proposed approach can be applied easily in real factory conditions and for large size problems. It should thus be useful to both practitioners and researchers.

Keywords: Scheduling, Overlapping, Job Shop, Simulated Annealing;

1. Introduction

The job shop scheduling problem is to determine a schedule of jobs that have pre-specified operation sequences in a multi-machine environment. In the classical job shop scheduling problem (JSP), n jobs are processed for completion on m unrelated machines. For each job, technology constraints specify a complete, distinct routing which is fixed and known in advance. Each operation uses one of the m machines for a fixed duration. Each machine can process at most one operation at a time and once an operation initiates processing on a given machine it must complete processing on that machine without interruption. The operations of a given job have to be processed in a given order. Each machine is continuously available from time zero, and operations are processed without preemption. taking into account the precedence constraints ,the problem consists of finding a schedule of the operations on the machines, which minimizes one objective such as make span $C_{\max}$ that is, time needed to finish the last operation in the schedule.

This paper considers job shop scheduling problems in a part making company. The company makes various parts and equipment for petrochemical industries. The company gets its orders at the beginning of each month. After receiving the orders, the company designs the operations specifications (the operations must be executed for each job, the capable machine for each operation and the process time for the capable machine of each operation) of the ordered jobs. This designing is done based on the received orders and the facility specifications of the company. The information on the jobs and their operations for each month is used in scheduling program proposed in this paper. The scheduling program searches the best sequence of the assigned operations on each machine confirming to an objective function. The scheduling problem considered in this paper is based on a job shop scheduling problem. In the classical job shop scheduling problem (JSP), each job has a demand that equals one. This assumption isn't true for many job shops such as job shops in petrochemical industries. In these job

[*] Corresponding author, Tel.: +98-811-8257410; Fax: +98-811-8257400; e-mail: Fattahi@basu.ac.ir.

shops and many of others, a demand received from a customer has a quantity more than one for each job. Demand determines the quantity of each finished goods required by a customer. In these job shops, the operations to be performed for each job can, and in fact should, overlap. The overlapping is limited by structural constraints, such as the dimensions of the box to be packed or the capacity of the container used to move the pieces, a product that is not completed, from one machine to the next. For this reason, we consider a job shop scheduling with overlapping problem and present an algorithm to solve it. The effects of the new approach on job shop scheduling problems are evaluated. Since the problem is well known as NP-Hard class, a simulated annealing algorithm is developed to solve large scale problems. Moreover, a mixed integer linear programming (MILP) method is presented to validate the proposed algorithm. The approach is tested on a set of random data to evaluate and study the behaviour of the proposed algorithm. So, firstly, validation of the SA algorithm is discussed and then the effect of the overlapping in job shop scheduling is evaluated.

Let us consider an example of two jobs and three machines job shop with overlapping problem given in table 1. In this problem, Job1 has three operations (named$o_{1,1}, o_{1,2}, o_{1,3}$) and job2 has two operations (named$o_{2,1}, o_{2,2}$). Job1 must produce 10 products of A and job 2 must produce 4 products of B. In this job shop we don't have to wait until all the $o_{1,1}$ operations have been done to start the $o_{1,2}$ operations. Figure 1 shows an illustration of overlapping in operations for job1. As shown in this figure, operations $o_{1,1}, o_{1,2}$ and $o_{1,3}$ are overlapped. Sometimes this overlapping is automatic, that is, as soon as the first piece has been processed on one machine it goes directly to the next machine. On other occasions the overlapping is limited by structural constraints, such as the dimensions of the box to be packed or the capacity of the container used to move the pieces from one machine to the next. We assume that each job has a due date and thus an attempt is made to meet as many due times as possible. A delay penalty is charged if a job is completed after the due tome promised to a customer. This penalty is assumed to be constant over time and includes cost of lost future sales or changed orders and rush shipping cost.

Table1

An example of two jobs and three machine job shop

Job name	Quantity	(machine sequence, Processing time)		
Job 1	10 A	(1,40)	(2,60)	(3,50)
Job 2	5 B	(3,60)	(2,40)	

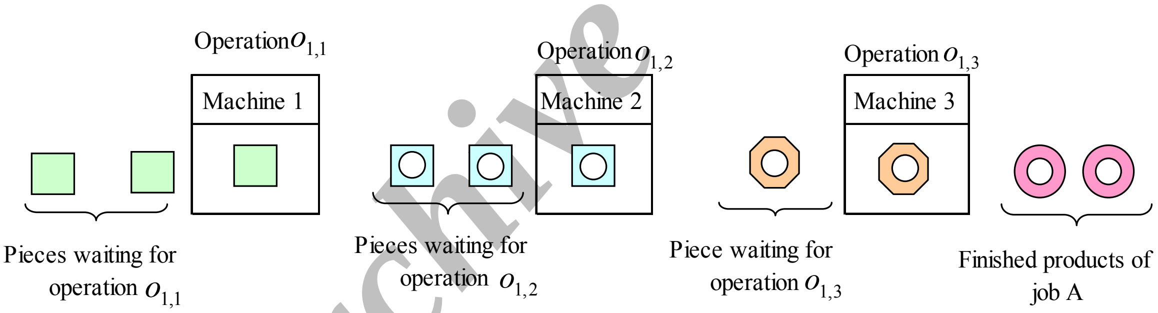
Operation$o_{1,1}$
Machine 1
Operation$o_{1,2}$
Machine 2
Operation $o_{1,3}$
Machine 3
Pieces waiting for operation $o_{1,1}$
Pieces waiting for operation $o_{1,2}$
Piece waiting for operation $o_{1,3}$
Finished products of job A

Fig. 1. An illustration of overlapping in operations

The job shop scheduling problem has attracted the attention of many researches. This problem consists of determining the production sequences and the starting times for a given set of non splittable jobs to optimize an objective function such as minimization of the makespan [1 and 9] or tardiness minimization [4]. The general JSP is strongly NP-hard [7]. For this reason many studies have focused on developing heuristic procedures for this problem. Historically JSP has been primarily treated using the following approaches: Branch and bound [3], heuristic procedures based on priority rules [8] and shifting bottleneck [6].

Problems of dimension "15×15" are still considered to be beyond the reach of today's exact methods. Over the last decade, a growing number of metaheuristics procedures have been presented to solve hard optimization problems: Simulated annealing [5], Tabu search [12], Neural network algorithms [10] and Genetic algorithms [13].

Saidi-Mehrabad and Fattahi [11] studied a job shop scheduling problem with lot-sizing consideration and an effective simulated annealing algorithm proposed to solve it. In their study the overlapping in operations is described as a new subject in job shop scheduling. The objective function considered in their evaluation is the minimization of the total weighted tardiness of jobs. They discussed that their proposed algorithm can obtain the good quality solutions in a small time. The effect of the proposed approach in job shop scheduling isn't studied in another literature. Job shop scheduling with overlapping in the operations, is a famous problem in the job shop industries such as job shops in petrochemical industries and others.

We developed a mixed integer linear program for this problem that can solve the small size problems in an algorithm for solving large problem instances. The approach is tested on a set of random data to evaluate and study the behaviour of the proposed algorithm. The computational results validate the effectiveness of the proposed algorithm. To evaluate the overlapping considerations on the job shop scheduling problem, we compare the results of classical job shop scheduling with the new approach (job shop scheduling problem with overlapping in operations). Three performance measures are used in this evaluation: Make span ($C_{\max}$), total weighted tardiness of jobs and total flow time.

The reminder of this paper is organized as follows: Section 2 describes the problem under consideration. The solution procedure is presented in section 3. Section 4 discusses numerical experiments and discussion. Section 5 includes concluding remarks.

2. Problem description and formulation

The scheduling problem under consideration has m machines and n jobs. The model proposed by Saidi-Mehrabad and Fattahi [12] is developed and used to represent a mathematical model of this problem. In this model, each job consists of a sequence of operations $O_{j,h}, h = 1,\ldots,h_j$, where $O_{j,h}$ and h_j denote that the operation of job j and the number of operations are required for job j, respectively. Unless stated otherwise, index i denotes a machine, index j denotes job, and h denotes operation throughout the paper. The due date d_j of job j is promised to its customer, and a delay penalty is charged if job j is completed after its due date. Let w_j denote the delay penalty per unit time of job j and n_j denote the demand of job j. The demand determines

acceptable time. It is well known that this problem is NP-hard. Therefore, we have developed a simulated annealing the quantity of a job required by a customer. Table 2 shows an example of two jobs and two machines problem. The operations of each job can overlap. The extent of this overlapping is limited by a coefficient, $ov_{j,h}$ defined as the proportion of operation $O_{j,h}$ that has to be processed before its successor $O_{j,h+1}$ can start. This coefficient is 1 if no overlapping is allowed and can be very low, nearly 0, if there is automatic overlapping, in which as soon as the first unit of operation $O_{j,h}$ is processed it becomes part of the next operation and is immediately processed. The overlapping, $ov_{i,j}$ of two operations determines the earliest starting time of the second operation, so that it is guaranteed that a sufficient proportion of $O_{j,h}$ is already processed and also that $O_{j,h+1}$ does not finish before the arrival of the last unit coming from $O_{j,h}$ (See Fig. 2).on the left hand in Figure 2, the first condition determines the earliest starting time of $O_{j,h+1}$. On the right hand in Figure 2, the second condition determines the earliest finishing time of $O_{j,h+1}$ [2].

Table 2
Example 2: Two jobs–two machines problem Operation sequence, machines alternative and processing time

Jobs	Quantity	Weight	Due date	Operations		
				1	2	3
Job1	100	0.3	7000	m_1,15	m_2,25	m_2,40
Job2	50	0.7	8000	m_1,10	m_2,14	m_2,30

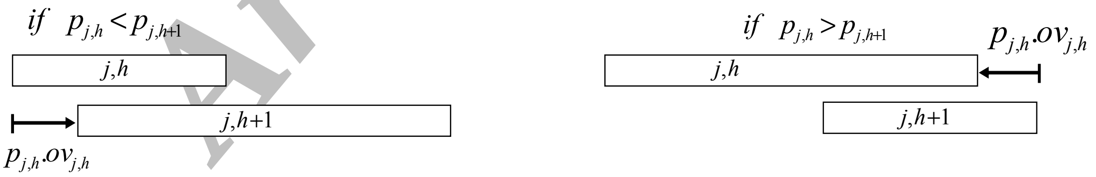

Fig. 2. Overlapping in operations

Under these assumptions and notations, the problem is to find a schedule that minimizes the objective measure (such as total weighted tardiness) given $n, m, O_{j,h}, a_{i,j,h}$ and $p_{j,h}$. (The following model is formulated for the total weighted tardiness objective and it can be transformed for other objectives easily)

$$a_{i,j,h} = \begin{cases} 1 \textit{ if } O_{j,h} \textit{ must be performed on machine i} \\ 0 \quad \textit{otherwise} \end{cases}$$

T: Total tardiness of schedule

M: A large number.

$$x_{i,j,h,k} = \begin{cases} 1 & \textit{if } O_{j,h} \textit{ is performed on machine i in priority k} \\ 0 & \textit{otherwise} \end{cases}$$

$t_{j,h}$: Start time of the processing of operation $O_{j,h}$.

$Tm_{i,k}$: Start of working time for machine i in priority k .

T_j : The tardiness of job j

A mixed integer program for the JSP with the proposed approach is then given as

$$\text{Min } T = \sum_{j=1}^{n} w_j T_j$$

s.t.

$$T_j \geq \max\left(\left(t_{j,h_j} + p_{j,h_j}\right) - d_j, 0\right) \quad for \quad j = 1,...n; \quad (1)$$

$$t_{j,h} + p_{j,h}.ov_{j,h} \leq t_{j,h+1} \quad for\ j = 1,...,n;\ h = 1,\ldots,h_j - 1; \ (2)$$

$$t_{j,h} + p_{j,h} \leq t_{j,h+1} + p_{j,h+1} - p_{j,h}.ov_{j,h}$$
$$for\ j = 1,...,n; h = 1,\ldots,h_j - 1 \quad (3)$$

$$Tm_{i,k} + p_{j,h}.x_{i,j,h,k} \leq Tm_{i,k+1}$$
$$for\ i = 1,..,m; j = 1,..,n; h = 1,..,h_j; k = 1,..,k_i - 1;\ (4)$$

$$Tm_{i,k} \leq t_{j,h} + (1 - x_{i,j,h,k}).M$$
$$for\ i = 1,..,m; j = 1,..,n; h = 1,..,h_j; k = 1,..,k_i;\ (5)$$

$$Tm_{i,k} + (1 - x_{i,j,h,k}).M \geq t_{j,h}$$
$$for\ i = 1,..,m; j = 1,..,n; h = 1,..,h_j; k = 1,..,k_i; \quad (6)$$

$$x_{i,j,h,k} \leq a_{i,j,h} \quad for\ i = 1,..,m; j = 1,..,n; h = 1,..,h_j; k = 1,..,k_i;\ (7)$$

$$\sum_j \sum_h x_{i,j,h,k} = 1 \ for\ i = 1,..m; k = 1,..,k_i; \quad (8)$$

$$\sum_i \sum_k x_{i,j,h,k} = 1 \ for\ j = 1,..n; h = 1,..,h_j;\ (9)$$

$$t_{j,h} \geq 0 \quad for \quad j = 1,\ldots,n; \quad h = 1,\ldots,h_j; \quad (10)$$

$$x_{i,j,h,k} \in \{0,1\} \qquad (11)$$

Constraint (1) determines the tardiness of each job. Constraints (2, 3) direct each job to follow a specified operation sequence and consider the overlapping in operations. Constraint (4) directs each machine to process one operation at a time. Constraints (5) and (6) direct each operation $O_{j,h}$ can start after its assigned machine is idle and previous operation $O_{j,h-1}$ is completed. Constraint (7) determines the machine for each operation. Constraint (8) assigns the operations to a machine and sequence assigned operations on all machines. Constraints (9) force each operation to be performed only on one machine and one priority. Results of $x_{i,j,h,k}$ yield an assignment each operation on a machine and sequence assigned operations on all machines. To explain the model better, the problem of example2 is solved by this model and its solution is presented in figure 3. This solution defines the sequence described in figure 4 and its schedule graph is shown in figure 5.

$$\left|\begin{array}{l} x_{1,1,1,2} = 1 \\ x_{1,2,1,1} = 1 \\ x_{2,1,2,2} = 1 \\ x_{2,1,3,4} = 1 \\ x_{2,2,2,1} = 1 \\ x_{2,2,3,3} = 1 \end{array}\right.$$

Fig. 3. The optimized solution for example2.

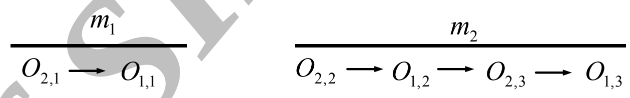

Fig. 4. Sequence of operations for Figure 3.

3. Solution procedure

3.1. Simulated annealing

Simulated annealing (SA) is a neighborhood search technique that has produced good results for combinatorial problems. A standard SA procedure begins by generating an initial solution at random. At each stage, the new solution taken from the neighborhood of the current solution is accepted as the new current solution if it has a lower or equal cost; if it has a higher cost, it is accepted with a probability that decreases as the difference in the costs increases and as the temperature of the method decreases. This temperature, which is simply a positive number, is periodically reduced by a temperature scheme, so that it moves gradually from a relatively high value to near zero as the method progresses. Thus, at the start of SA most deteriorating moves are accepted, but at the end only ameliorating ones are likely to be accepted. The method

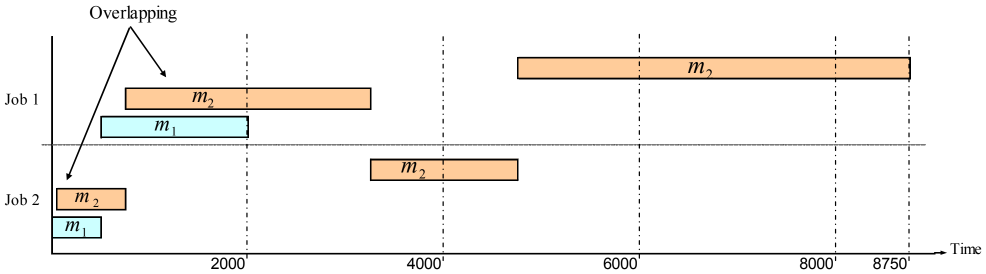

Fig. 5. The schedule graph of optimized solution for example 2

Converges to a local optimum as the temperature approaches zero, but because SA has performed many perturbations at higher temperatures which have pushed the search path into new areas, a better local optimum solution should hopefully be reached. In this paper, additional termination criteria are introduced into the proposed SA procedure to reduce the computational effort. The entire SA procedure is terminated either when the temperature$T$ reaches a pre-specified valueT_f or the frozen counter f_c is greater than a pre-defined value. In addition, to intensify and diversify the SA searching procedure, when a solution is not ameliorated in a pre-defined number of consecutive temperature stages, a "restarting solution" mechanism is designed to generate a new schedule S for the further amelioration of the solution.

3.2. The proposed algorithm

The scheduling problem of a JSP consists of sequencing the operations on all machines in order to obtain a feasible schedule minimizing the predefined objective function. In this section, we present an SA algorithm for determining the near optimal solution to minimize the predefined objective function. The elements of the SA algorithm are described as follows. In this paper, a matrix$S_{4\times w}$ is used to represent the solution seed. Index w denotes the total number of operations for all jobs. The matrix rows are described in figure 6. Row 4 determines the operations sequences on each machine. The initial sequence is obtained with a random operator that sequences the operations of each machine randomly. An initial sequence for example 2 is represented in figure 7.

$$\begin{array}{c|c|}
 & 1,\ldots,v \\
1 & \text{Machine number} \\
2 & \text{Job number} \\
3 & \text{Operation number} \\
4 & \text{Priority}
\end{array}$$

Fig. 6. The solution seed structure

$$\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array}\begin{bmatrix} 1 & 1 & 2 & 2 & 2 & 2 \\ 1 & 2 & 1 & 1 & 2 & 2 \\ 1 & 1 & 2 & 3 & 2 & 3 \\ 2 & 1 & 2 & 3 & 1 & 4 \end{bmatrix}$$

Fig. 7. An initial solution for example 2.

Given a matrix$S$, a neighborhood matrixS^c is obtained for$S$ using the following method: randomly select one machine and one priority of this machine. A neighborhood of$S$, S^c is obtained by pair wise interchanging priority number of a pair wise operations. A neighborhood matrix for the sequence presented in figure 7 is presented in figure 8. Briefly, the scheduling SA algorithm can be summarized with the pseudo-code in figure9. To make the performance of the proposed SA algorithm more robust, parameter setting is necessary. In this research the parameter setting for the designed SA heuristic is stated in detail. From the results of some preliminary experiments, the initial temperature ranges from 100 to 300, depending on the size of the problem concerned and the final temperature is determined to be 0.1 for all the cases of the problem. A temperature reduction factor r of 0.98 has been chosen. The maximum iteration length at a particular temperature Max_Iter is set to be 50 to 300.

$$\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array}\begin{bmatrix} 1 & 1 & 2 & 2 & 2 & 2 \\ 1 & 2 & 1 & 1 & 2 & 2 \\ 1 & 1 & 2 & 3 & 2 & 3 \\ 2 & 1 & 2 & 4 & 1 & 3 \end{bmatrix}$$

Fig. 8. A neighborhood for the initial sequence in figure7.

4. Numerical experiments and discussion

This section describes the computational tests which are used to evaluate the effect of the proposed approach: job shop scheduling with overlapping operations. Various problems of job shop scheduling generated randomly are used to evaluate the effect of the proposed approach. To solve the test problems, the proposed algorithm is coded by the visual Fortran language. Firstly, we evaluate the performance and effectiveness of the proposed algorithm and then we use the algorithm to evaluate the effect of job shop scheduling with overlapping operations.

To evaluate the performance and effectiveness of the proposed algorithm, it is tested in comparison with the traditional optimization technique (the branch and bound method). The MILP model (presented in section 2) is coded by Lingo software which solves the model by the branch and bound method. The proposed algorithm and the Lingo software were run on a PC operated by a Pentium-IV 1.89 GHz processor, with 512 Mb RAM. The test problems are solved by the branch and bound method with make span criterion and their results are shown in table3. This table shows the test problems specification, the results of the branch and bound method and the results of the proposed algorithm. A review of the results in table3 shows that, the proposed algorithm is capable of obtaining the near-optimal solution. Therefore, the proposed algorithm is effective for the problems considered here. In addition, the CPU times for the proposed algorithm are very smaller than the branch and bound method.

To evaluate the effect of the new approach, three criteria of job shop scheduling objectives are used: maximum completion time or make span ($C_{\max}(\sigma)$), total weighted tardiness ($\sum_{j=1}^{n} w_j T_j(\sigma)$) and total flow time. The test problems are solved by the proposed algorithm for each criterion and the optimal solutions are obtained. The results of the test problem with and without overlapping consideration are obtained and shown in table 4. Effect of overlapping consideration on job shop scheduling is shown in figures 10, 11, 12 and 13. Figures 10, 11 and 12 show the number of criteria in job shop scheduling with the classic and proposed approach for each test problem. The effect of the proposed approach on job shop scheduling is shown in figure 13. This figure shows that on average the proposed approach can improve 18% of the Make span criterion, 36% of the total weighted tardiness criterion and 21% of the total flow time criterion in job shop scheduling problems. Therefore, we conclude that the proposed approach can improve the criteria and machines utilization measures in job shop scheduling.

```
Step 1: Initialization
    Step 1.1: Obtain an initial sequence S.
    Step 1.2: Initiate the initial temperature $T$ , final temperature $T_f$ , and cooling rate $r$ .
Step 2: While not yet frozen, ($T > T_f$ ) or ($n_{bsc} < 20$), do the following.
    Step 2.1: Perform the following loop L times.
        Step 2.2.1: Neighborhood search. Select a neighbor $S^c$ of $S$ .
        Step 2.2.2: Compute  $\Delta = R_{S^c} - R_S$  (done sequencing algorithm)
        Step 2.2.3: If $\Delta \leq 0$ then $S = S^c$.
        Step 2.2.4: Compute $\Delta_b = R_S - R_{S^*}$
        Step 2.2.5: If $\Delta_b < 0$ , set $S^* = S$ and $n_{bsc} = 0$ .
        Step 2.2.6: If $\Delta > 0$ , select a random variable $P \sim U(0,1)$ .
                If $e^{-\Delta/T} > P$ , set $S = S^c$ .
    Step 2.2: Set $T = r \times T$.
Step 3: Return the best solution found for $S^*$ .
```

Fig. 9. The pseudo-code for the proposed algorithm

Table3

Comparing the proposed algorithm with the branch and bound method

Problem no.	Size	Results of the branch and bound method					Results of the proposed algorithm	
		Integer variables	Non integer variables	Number of constraints	CPU Time (Sec.)	$C_{\max}$	CPU Time (Sec.)	$C_{\max}$
JSPO1	2.2.2	18	18	71	2	2670	4	2670
JSPO2	3.2.2	36	25	154	2	3270	4	3270
JSPO3	3.3.2	90	38	340	4	4270	4	4270
JSPO4	3.3.3	81	27	331	4	2830	4	2830
JSPO5	4.3.3	180	52	585	12	3300	4	3300
JSPO6	4.3.3	144	49	585	9	3100	4	3100
JSPO7	5.3.3	225	61	911	393	3300	4	3300
JSPO8	5.3.3	225	61	911	50	3700	4	3700
JSPO9	5.3.4	300	66	896	70	3060	4	3060
JSPO10	5.4.4	680	80	1596	1170	3650	4	3700
JSPO11	5.4.4	640	88	1596	3600	3440,4670	4	4670
JSPO12	6.4.4	768	99	2299	3600	1830,6040	4	5710
JSPO13	6.4.4	672	95	2299	3600	1899, 5710	4	5710
JSPO14	6.4.5	720	97	2275	3600	2211,5640	4	5490
JSPO15	7.4.5	980	113	3102	3600	2360,-----	4	6060

5. Conclusion

In this paper, a new approach: job shop scheduling with overlapping in operations is presented and discussed. The proposed approach is useful in many job shops such as job shops in petrochemical industries. Since the considered problem is well known as NP-Hard class, a simulated annealing algorithm is developed to solve large scale problems. Moreover, a mixed integer linear programming (MILP) method is presented applied for small size problem to validate the proposed algorithm. The proposed approach implemented using the Visual Fortran language. Computational experiments confirmed superiority of the proposed approach and validate the efficiency and effectiveness of it. To show the effect of proposed approach in job shop scheduling problems, we compare the results of classical job shop scheduling with the proposed approach. Computational experiments show that averagely the proposed approach can improve 18% of the Makespan criterion, 36% of the total weighted tardiness criterion and 21% of the total flow time criterion in job shop scheduling problems. So it is concluded that the proposed approach can improve the criteria and machines utilization measures in job shop scheduling. Although the proposed approach is used in job shop scheduling, it can be easily adapted for scheduling problems such as flow shop and open shop scheduling problems which can be suggested for future research.

Table4

Effect of overlapping consideration on objective criteria

Problem no.	Size	Makespan criterion			TWT criterion			TFT criterion		
		With overlapping consideration	Without overlapping consideration	Percent	With overlapping consideration	Without overlapping consideration	Percent	With overlapping consideration	Without overlapping consideration	Percent
JSPO1	2.2.2	2670	3300	0.81	1030	1750	0.59	3960	5400	0.73
JSPO2	3.2.2	3270	3900	0.83	1013.3	1733.3	0.58	6810	8700	0.78
JSPO3	3.3.2	4270	5270	0.81	1643.3	2313.3	0.71	9630	11720	0.82
JSPO4	3.3.3	2830	3700	0.76	1275	1963.3	0.65	7225	9290	0.77
JSPO5	4.3.3	3300	3600	0.91	981.25	1677.5	0.58	8230	9770	0.84
JSPO6	4.3.3	3100	3920	0.76	1068.75	1777.5	0.60	9060	10820	0.84
JSPO7	5.3.3	3300	3970	0.83	980	1304	0.75	12030	14590	0.82
JSPO8	5.3.3	3700	4300	0.86	1041	1560	0.66	10820	13150	0.82
JSPO9	5.3.4	3060	3800	0.80	882	1530	0.58	9650	12500	0.77
JSPO10	5.4.4	3650	4890	0.74	1195	1726	0.69	13470	17920	0.75
JSPO11	5.4.4	4670	5690	0.82	1383	2038	0.68	15420	19400	0.79
JSPO12	6.4.4	5710	6340	0.90	2288.3	3356.6	0.68	21810	27150	0.80
JSPO13	6.4.4	5710	6300	0.90	1355	2091.6	0.64	21985	28000	0.78
JSPO14	6.4.5	5490	6300	0.87	1160	1880	0.61	18135	22570	0.80
JSPO15	7.4.5	5560	8260	0.67	1324.3	2554.3	0.52	23625	31670	0.74

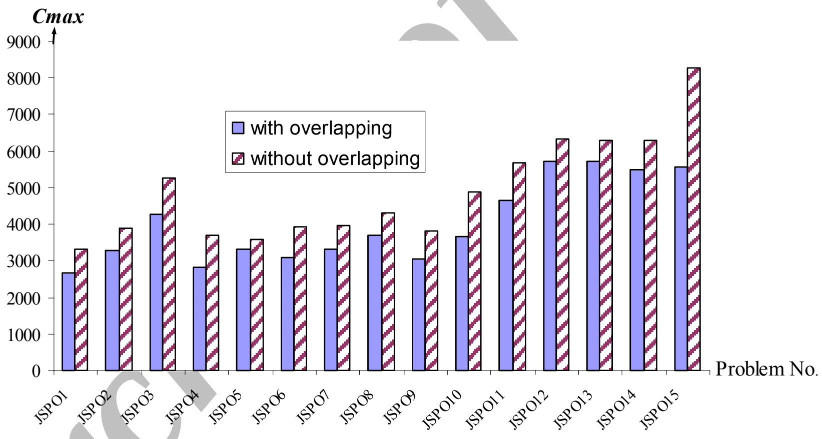

Fig. 10. Effect of the proposed approach on job shop scheduling with Cmax criterion

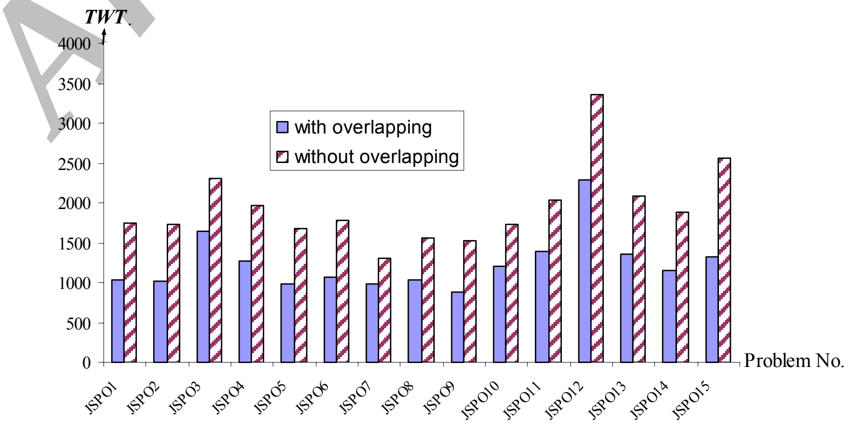

Fig. 11. Effect of the proposed approach on job shop scheduling with TWT criterion

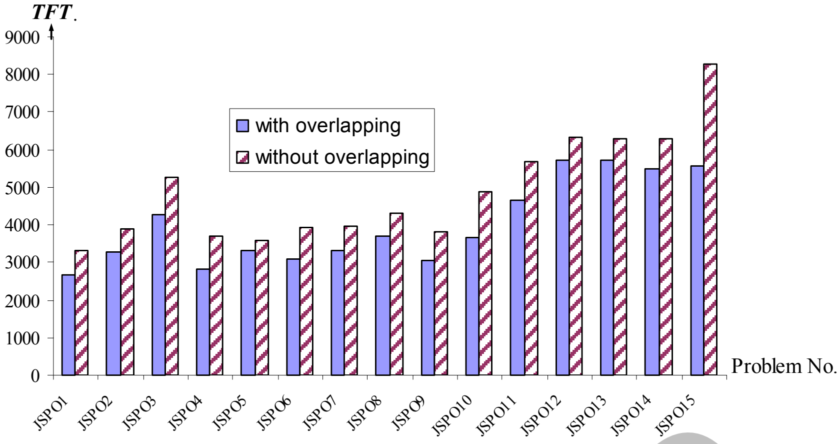

Fig. 12. Effect of the proposed approach on job shop scheduling with TFT criterion

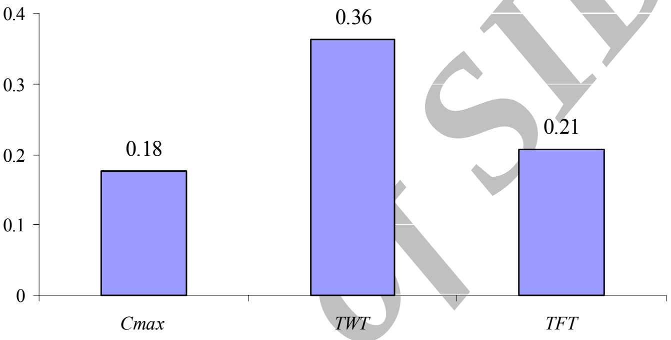

Fig. 13. The average improvement of job shop scheduling criteria with the proposed approach

Reference

[1] J. Adams, E. Balas, D. Zawack, The shifting bottleneck procedure for job shop scheduling. Management Science, 34, 391–401, 1988.

[2] V. Alvarez, A heuristic to schedule flexible job shop in a glass factory European. Journal of Operation Research, 165, 525–534, 2005.

[3] C. Artigues, M. Gendreau, L. Rousseau, A. Vergnaud, Solving an integrated employee timetabling and job-shop scheduling problem via hybrid branch-and-bound. Computers & Operations Research, 36, 2330-2340, 2009.

[4] K. R. Baker, Sequencing rules and due-date assignments in a job shop. Management Science, 30, 1093–1104, 1984.

[5] E. Bouri, N. Azizi, S. Zolfaghari, A comparative study of a new heuristic based on adaptive memory programming and simulated annealing: The case of job shop scheduling. European Journal of Operational Research, 177, 1894-1910, 2007.

[6] J. Gao, M. Gen, L. Sun, X. Zhao, A hybrid of genetic algorithm and bottleneck shifting for multi objective flexible job shop scheduling problems. Computers & Industrial Engineering, 53, 149-162, 2007.

[7] M. R. Garey, D. S. Johnson, R. Sethi, The complexity of flow shop and job shop scheduling. Mathematics of Operations Research. 1, 117-129, 1976.

[8] C. Gray, M. Hoesada, Matching heuristic scheduling rules for job shops to the business sales level. Production and Inventory Management Journal, 4, 12–17, 1991.

[9] P. V. Laarhoven, E. Aarts, J. Lenstra, Job shop scheduling by simulated annealing. Operation Research. 40, 113-125, 1992.

[10] J. S. Meeran, Job shop scheduling using neural networks. International Journal of Production Research, 36, 1249-1272, 1998.

[11] M. Saidi Mehrabad, P. Fattahi, Job shop scheduling with lot-sizing considerations. 36th CIE Conference on Computer & Industrial Engineering, Taipei, Taiwan , 440-445, 20 –23 June., 2006.

[12] M. Saidi Mehrabad, P. Fattahi, Flexible job shop scheduling with tabu search algorithm. International Journal of Advanced Manufacturing Technology, 32-563-570, 2007.

[13] H. Zhou, W. Cheung, L. Leung, Minimizing weighted tardiness of job-shop scheduling using a hybrid genetic algorithm. European Journal of Operational Research, 194, 637-649. 2009.