International Journal of Mechanical, Industrial and Aerospace Sciences
ISSN: 2517-9950
Vol:1, No:2, 2007

Multiple Job Shop-Scheduling using Hybrid Heuristic Algorithm

R.A.Mahdavinejad

***Abstract*—**In this paper, multi-processors job shop scheduling problems are solved by a heuristic algorithm based on the hybrid of priority dispatching rules according to an ant colony optimization algorithm. The objective function is to minimize the makespan, i.e. total completion time, in which a simultanous presence of various kinds of ferons is allowed. By using the suitable hybrid of priority dispatching rules, the process of finding the best solution will be improved. Ant colony optimization algorithm, not only promote the ability of this proposed algorithm, but also decreases the total working time because of decreasing in setup times and modifying the working production line. Thus, the similar work has the same production lines. Other advantage of this algorithm is that the similar machines (not the same) can be considered. So, these machines are able to process a job with different processing and setup times. According to this capability and from this algorithm evaluation point of view, a number of test problems are solved and the associated results are analyzed. The results show a significant decrease in throughput time. It also shows that, this algorithm is able to recognize the bottleneck machine and to schedule jobs in an efficient way.

***Keywords*—**Job shops scheduling, Priority dispatching rules, Makespan, Hybrid heuristic algorithm.

I. Introduction

SOLUTION to a job shop-scheduling problem is a sequence of jobs to be processed on machines in alternative routes. The numerous variables and constraints are involved in job shop scheduling. Its complexity of the large solution space and its multi-criteria objective function make the problem difficult. This problem is a class of NP-Hard ones that cannot be optimally solved for large-scale problems in a reasonable amount of computational time. For this reason, great deals of researches develop and work on heuristic methods to find near-optimal solutions. In general to solve such problems, typical methods are introduced as: local searching algorithm [1], priority dispatching rules [2], expert systems [3], ant colony optimization [4], genetic algorithms [5], branch-and-bound algorithm [6], shifting bottleneck algorithm [7], tabu search [8], simulated annealing [9], neural network [10] and graph theory [11]. In addition to the above methods, a hybrid method can be also proposed such as: local search and simulated annealing algorithms [12], fuzzy and priority dispatching rules algorithms [13], local search and genetic algorithms [14].

During 1950s, many problems were solved by efficient heuristics taken based on classic scheduling developments. Various priority rules have been established [Panovaker & Eskandar 1977]. The researches in this field show that the best methods are proposed by the combination of some priority dispatching rules [Fisher & Thompson 1963, Laurence 1984]. Due to the weakness of these priority rules, there is still a special attraction to develop and propose more efficient methods. Fisher and Renoykan [1988] have first studied in this field. Roudamber and White [1988] have shown that the ability and flexibility of these methods are much better than the optimization methods. From 1988 to 1991, a shifting bottleneck method was proposed and it was a very significant change in approximate methods, which was the first heuristic method for solving FT10 problems [Adoms, *et. al.* 1988]. Simulated annealing (SA) method is a well-known meta-heuristic method that is widely used in job shop scheduling problems. SA cannot find the exact solution for large-scale problems with high speed rate. So to improve this weakness, this method can be combined with some other methods such as critical neighborhoods to produce active job shop scheduling [Yamada 1995 and 1996] and genetic algorithms to speed up the process efficiently [Koloneko 1998].

II. Mathematical Model

There are some assumptions in job shop scheduling listed bellow:

- At the time beginning, all workstations are ready to start and process jobs.
- Machines are arranged according to the functional/process layout based on similar functions in the job shop. For example, turning machines are in one group, milling machines in another group and so on. In this paper, each machine in these groups is called workstation.
- Each job is allowed to visit any workstation, which is determined by the algorithm proposed in this paper. This assumption exists to the most real-world situation. Note that the total time, which includes of setting and processing time on jobshop places, because of the ability differences in working stations, can be different from each other. Besides, number of working places in production line and also the number of working stations in a working place are as input parameters.

Author is with the School of Mechanical Eng., Eng. Faculty, University of Tehran, Tehran, Iran. E-mail:

- Every job in jobshop is considered as a non-natural ant.
- If there are various kinds of jobs in jobshop at the same time, then every job will remain its own fermon in selecting its moving line, so that it will also pay only attention to its own fermon in selecting its moving trace. In this paper, ant colony optimization algorithm has been used. So that, depending on the numerous jobs, various kinds of fermons have been considered for non-natural ants instead of one special fermon.
- Every job can pass through a special working place only once during its production sequences. So coming back to a special working place is forbidden according to this assumption there is not any necessarily to create a special forbidden list for every non-natural ant. This list without the assumption mentioned is quite necessary to correct the action of the ant colony optimization algorithm.

The variables and parameters of this model are as follows:

N_i^j = Number of remining process for job *j* at the stage of scheduling that this job can be assigned to station *i*.

$Y_{i,j}$ = Ready time for processing job *j* on station *i*.

$\pi_{i,k}^j$ = Amount of feremon remined from job j on the route between station *i* to station *k*.

$\tau_{i,k}^j$ = Severity of feremon remined from job j on the route between station *i* to station *k*.

The mathematical model of a muli-job shop scheduling process is as follows:

Minimize C_{max}

S.t.

$$Y_{kj} - Y_{ij} \geq P_{ij} + \lambda_j . S_{ij} \qquad \forall (i,j) \rightarrow (k,j) \qquad (1)$$

$$C_{\max} - Y_{ii} \geq P_{ii} + \lambda_i . S_{ii} \qquad \forall (i,j) \qquad (2)$$

$$\left\{ \begin{array}{l} Y_{ij} - Y_{il} \geq P_{il} + \lambda_l . S_{il} \\ \qquad\qquad \text{or} \\ Y - Y \geq P + \lambda . S \end{array} \right. \qquad \forall (i,l),(i,j) \qquad (3)$$

$$\left\{ \begin{array}{l} Y_{ij} - Y_{kj} \geq P_{kj} + \lambda_j . S_{kj} \\ {}_{ki} \quad {}_{ii} \quad {}_{ii} \quad {}_{j} \quad {}_{ij} \end{array} \right. \qquad \begin{array}{l} \forall (i,j),(k,j) \qquad (4) \\ \forall (i,j) \qquad (5) \end{array}$$

$$P_{ij}, S_{ij} > 0 \quad {}_{ij} \qquad \forall (i,j) \qquad (6)$$

$$1 \geq \lambda_j > 0 \qquad \forall j \in \{1, \dots, m\} \qquad (7)$$

III. Proposed Algorithm

In this section, the priority algorithm will be discussed step by step. First of all, it is necessary to give some more explanations about this algorithm. In this work, the scheduling jobs are as unreal ants. These ants are doing making decision to find the traces, which are work circumstance. These decisions are based on the selection of a rule, which is according to the action of two kinds of information. The first one as general information is those, which are attending to experience of other ants their tracing lines. Localized information is the second one, which is concerned to the decision of ant to travel from one working station to the next one. The first three sentences of equation is the type of this kind of information. Optimization is doing on general information, as localized information refers to the specification of the job at any point. So that these specifications are a part of job and making any difference will be as revolution of the subject itself. Multiprocessor job shop scheduling with any algorithm with the aim of duration time optimization has to optimize the setting times on each working station. Because due to the variety and low rate productions in job shop, the general machines usually without any special accessories as fixtures are used. In this manner, working setup is easy. On the contrary, in much production, the machine is designed for a special job with easy and rapid function. From this point of view, there is a noticeable difference between job shop with other production process in the time, which is needed to set up the workplace and machine itself. Therefore, optimization of scheduling job shop and minimization of setting times is very important. So in this research multi fermons has been considered for various kinds of jobs in job shop. In this manner, every job in jobshop is to be concerned to an unreal ant. A special unreal ant is also considered to a special job. Fermon of an ant is different from the others. So that the fermons of two different ants pass through a special trace (line) can't be gathered together. An ant can follow its line only by the fermon of its own kind, which is remained and will not follow any traces with other kinds of fermons.

This is the same as waves tracing (following) model with various kinds of wavelength, which is introduced by Zinkler and Warela [15]. Sensibility of unreal ants only to their own fermons is the trends by which, the setting times on working station can be minimized. Supposed two kinds of jobs (1) and (2) can be done on a special working station. If you are going to do the job (2) on the working station that is doing job (1), then you need an extra time to prepare and set system up. But reprocessing of job (1) on this workstation will save a noticeable set up time. So it is better to process similar jobs on a special working station. In this work, as classical optimization algorithm of ant colony, three steps: The calculation of a trace selection probability, Selection of a trace among others a finally updating of fermons on all traces are considered. The sequences of proposed algorithm are as follows:

- **First step:** Formation of schedulable processes.
- **Second step:** Calculation of unreal processing times for all schedulable processes on their working stations. In this scheduling unreal time will be the start of processing time if the processing allocates to considered station.

- **Third step:** The allocation probability will be calculated for all scheduling processes. $\pi^{C}_{a,b}$ is defined as the fermon amount related to work (c) which is remained on line from (a) to (b) working stations, so that:

$$\tau^{C}_{a,b} = \frac{\pi^{C}_{a,b}}{\sum_{k,i} \pi^{K}_{i,b}} \tag{8}$$

$\tau^{C}_{a,b}$ Is fermon intensifying from unreal ant (c) remained on line working stations from (a) to (b).

$\sum_{k,i} \pi^{K}_{i,b}$ Is the assumption of fermons remained of whole unreal ants on working stations from (a) to (b).

In this priority algorithm the following equation is used to calculate the allocated probability of processes:

$$p^{C}_{a,b} = \frac{\left(\frac{1}{T^{C}_{b}}\right)^{l} \cdot \left(R^{C}_{b}\right)^{m} \cdot \left(\frac{1}{1+S^{C}_{b}}\right)^{n} \cdot \left(\tau^{C}_{a,b}\right)^{p}}{\Sigma \left(\frac{1}{T^{k}_{b}}\right)^{l} \cdot \left(R^{k}_{b}\right)^{m} \cdot \left(\frac{1}{1+S^{k}_{b}}\right)^{n} \cdot \left(\tau^{K}_{i,b}\right)^{p}} \tag{9}$$

The parameters of this equation are as:

$p^{C}_{a,b}$ = Movement probability of work (c) from (a) to (b) stations.

T^{C}_{b} = Processing time of work (c) on station (b).

R^{C}_{b} = Remained relative working time of work (c) processing on work station (b).

S^{C}_{b} = Assumed start processing time of work (c) on station (b).

l,m,n,p = Constant parameters which can be any real and positive number.

These parameters evaluate the importance coefficient of processing times remained relative time, assumed start processing time and remind intensify of fermons on each line.

K: scheduling processes that are going to be process on (b) working station.

T^{C}_{b} can be determined from following equation:

$$T^{C}_{b} = TP^{C}_{b} + \lambda^{C} . TS^{C}_{b} \tag{10}$$

so that:

TP^{C}_{b} = Processing time of work (c) on (b) working station.

TS^{C}_{b} = Setup time of work (c) on (b) working station.

λ^{C} = Input coefficient between zero and one which is used in any work. This coefficient is applied on times needed to set the piece up, when the working station had been processing the similar work. Otherwise this parameter will be equal to one. λ^{C} shows saving times on working stations with repeated processing indeed. R^{C}_{b} can also be calculated as:

$$R^{C}_{b} = \frac{N^{C}_{b}}{\Sigma N^{k}_{b}} \tag{11}$$

N^{C}_{b} is the remained processing number of (c) work before processing on (b) working station.

- **Fourth step:** After calculation of allocating probability of each process of schedulable processes set in third step this amount will be order descending.
- **Fifth step:** The allocation of schedulable processes to working stations will be with the accordance of their highest priorities.
- **Sixth step:** After allocation process in fifth step, allocating probability of all processes, which are in schedulable processes set and will allocate to allocated work station in fifth step, become zero.
- **Seventh step:** Starting and processing duration and also final production time will be calculated in this step.
- **Eighth step:** Return to fourth step until the allocation probability of whole schedulable processes is zero.
- **Ninth step:** Return to first step and making a new collection of schedulable process until the new collection is empty.
- **Tenth step:** Record the longest completion time for scheduled processes as make span. In this step a cycle will be completed.
- **Eleventh step:** At the end of a cycle the fermons of various unreal ants on all traces must be up to date. As mentioned before, there may be more than one job on production line so in spite of their similarity; they may travel through various lines. So the question is that, for a special job, on which line the fermon will be strengthened or weakened. To answer this question, it is noticeable that among similar lines, the fermon will be strengthened on the shortest one. Up dating values of fermons is done by following equation:

$$\pi^{C}_{i,j}[t+1] = \begin{cases} (1-\rho).\pi^{C}_{i,j}[t] & \text{(b)} \\ R + (1-\rho).\pi^{C}_{i,j}[t] & \text{(a)} \end{cases} \tag{12}$$

The equation (12a) is valid only when the selected unreal ant (c kind) has not passed through line *I* to *j* otherwise the equation (12b) will be satisfied. In this equation input values of ρ and *R* are fermon evaporation rate (0-1) and real value greater than one respectively.

- **welfth step:** Referring to the first step and starting the next optimization cycle until coming into the end conditions is satisfied. These conditions can be the optimization of time to solve the problem, the optimization number of cycles or percentage value optimization and so on.

- **Thirteenth step:** Introducing the least amount of total time resulting from optimization cycles during production.
- **Fourteenth step:** the end of algorithm.

The flowchart of this algorithm is shown in figure 2.

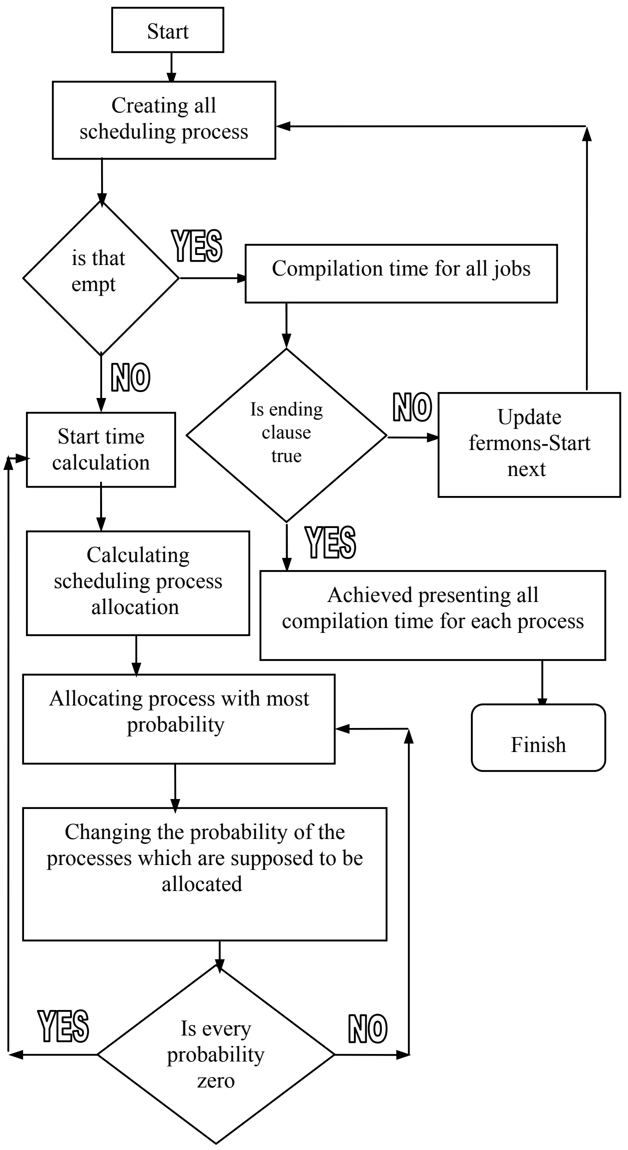

Start
Creating all scheduling process
is that empt
YES
Compilation time for all jobs
NO
Start time calculation
Is ending clause true
NO
Update fermons-Start next
YES
Calculating scheduling process allocation
Achieved presenting all compilation time for each process
Allocating process with most probability
Finish
Changing the probability of the processes which are supposed to be allocated
YES
Is every probability zero
NO

Fig. 1 Proposed algorithm to solve multi-processor job shop scheduling problem

The item $(1/T)$ in eq.9 denotes the use of preference distribution law for the shortest processing time. Using this law minimizes the handling time of pieces in jobshop and after that decreasing of total completion time. But this law has also a weakness. Suppose a special job with long process time is together with some other jobs short process times that are to be processed. In this manner, the job with longest processing time will always waiting at the end of line. According to this law to compensate this weakness, the term relative remained time R is considered in this equation. So with this term the arrangement of the jobs with long processing times will be according to their remained processing amounts and this is obviously a better solution of scheduling job shop problems. The term $(\frac{1}{1+S})$ is also considered the waste times on working stations. Therefore, the allocation preference in the same conditions will be to process with the earliest starting of unreal processing time. The term $\tau^{C}_{a,b}$ in each step of scheduling causes the selection of the highest density of fermon among working stations. This means that similar jobs will be processed on a special working station, so the setup times will be minimized by not processing of jobs on the other working stations.

IV. Discussion

The general specifications of selected problems to priority algorithm evaluation are shown on table 1.

The results of priority and other compared algorithm, solving time and also the answering deviations of problems from their lower limits are shown in figures 2 and 3 respectively. Among these algorithms, MS and LPT have the similar behavior. CR and SPT algorithm show also better operations in increased dimensions (see 6*6*6 results in Fig. 3). But priority algorithm shows a uniform and predictable behavior. It also reaches to a constant amount at high values, which is the reason of its good operation at high dimensional problems. The results of these figures are obtained when *l,m,n*=1.c since, the best answering of priority timed algorithm is when processing relative remind and starting unreal processing times are at the same rank of importance. So this can be convenient suggestion value to these parameters.

TABLE I
INPUT DATA FOR THE FIRST TEST PROBLEM

Work Counter	Work station	Setup time	Processing time
(1,1)	(1,1,1,2)	(3,1,2,3,)	(2,1,2,3)
	(1,2,1,3)	(3,2,1,3)	
	(1,3,2,2)		
(1,2)	(1,1,1,2)	(3,1,2,3,)	(2,1,2,3)
	(1,2,1,3)	(3,2,1,3)	
	(1,3,2,2)		
(2,1)	(1,1,1,3)	(3,1,1,2)	(2,1,2,2)
	(1,2,2,2)	(3,2,2,3)	
	(1,3,2,3)		
(3,1)	(1,1,2,3)	(3,1,2,2)	(2,1,1,3)
	(1,2,3,2)	(3,2,2,3)	
	(1,3,4,2)		

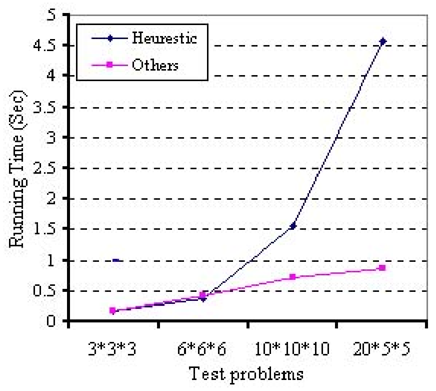


Fig. 2 a comparison of execution time between proposed algorithm and the others

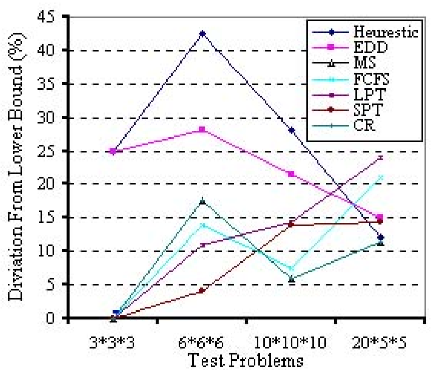


Fig. 3 a comparison of the deviation between proposed algorithm and the others

Fig. 4 shows the result for typical problem with different values of *l, m* and *n.*

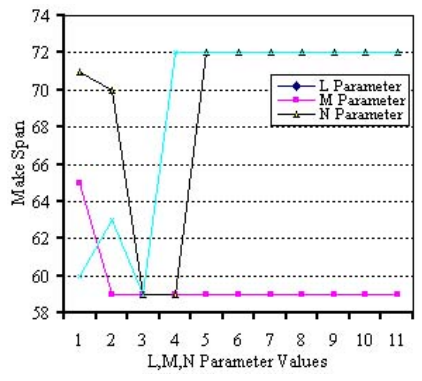


Fig. 4 sensivity analysis of parameters *l*, *m*, and *n* for a single-processor (6*6*6)

There is various factors in multiprocessing of job shop scheduling problems which make them complex in comparison with single processors. So the priority algorithms in this range are less suggested with their own assumption and in comparison between these algorithms the similarity assumption must be under attention. The evaluation of priority multi processor algorithm is taken by two typical examples. These examples are designed so that each of them shows some capability of priority algorithm. On the other hand this priority multi processor indeed, so its operation somehow can be trusted.

The first example is the factory with three working positions in which there are three, two and one stations at these one to three positions respectively. It is also assumed that three kinds of jobs, from the first kind and one job from the second and third kind are considered. More details of these examples are mentioned in table 1. The input parameters of this problem are as follows:

The primary value of fermon related to various unreal ants on all lines (π_0) is considered to be 5 (see eq.8).

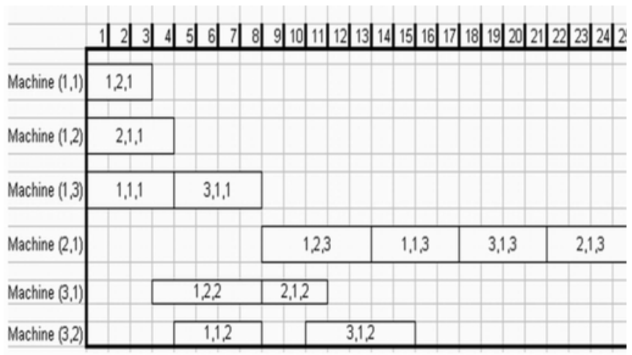
The coefficient λ is considered 0.5 for any job (see eq.10).

The fermon evaporation rate (ρ) is to be 0.5 (see eq.12).

The values of *l, m, n and p* parameters are considered equal to 1 (see eq.9).

This problem is solved by Pentium 4 with clock 4.2 GHz CPU on 0.27 second and the answering is 35. Table 2 is the Gantt chart of this example.

TABLE II GANTT CHART FOR THE FIRST TEST PROBLEM OF THE MULTI-PROCESSOR



As it is shown in this table station 2 is the bottleneck of this example due to unique working station. The specializing processes to this working position there are two main points from focus point of view; 1)The job has been specialized to this working station as soon as possible (t=8). This means that these preprocess have been scheduled so that the specialized processing has been done at the shortest time. 2)As a bottle neck working station after the first specializing till the end of scheduling time other processes are specialized to this station so the there is not any wasted time on it. This condition is optimized specialized process situation for any working station.

Another example is an objective problem to show the using gain of multi fermons on scheduling quality of flexible job shop production with priority algorithm. Consider a factory whit two working positions and two working station in each position. There are two kinds of work piece and two pieces of each one in the factory. The details are shown in table 2.

TABLEIII INPUT DATA FOR THE SECOND TEST PROBLEM

Work type,Work counter	Work place,Work station,Setup,Processing time	
(1,1)	(2,1,1,5)	(1,1,2,3)
	(2,2,1,6)	(1,2,3,4)
(1,2)	(2,1,1,5)	(1,1,2,3)
	(2,2,1,6)	(1,2,3,4)
(2,1)	(2,1,1,5)	(1,1,3,2)
	(2,2,1,6)	(1,2,3,3)
(2,2)	(2,1,1,5)	(1,1,3,2)
	(2,2,1,6)	(1,2,3,3)

This problem is solved with the parameters as follows the primary value of fermon related to various unreal ants on all lines (π_0) is considered to be 5 (see eq.8).

The value of coefficient λ is to be 0.3 (see eq.10).

The evaporation rate of fermon (ρ) is considered to be 0.5 (see eq.12).

The parameters *l,m,n,p* are to be equal to one (see eq.9).

This problem is solved by Pentium 4 with clock 4.2 GHz CPU on 0.12 second and the answering is 17.2.

V. RESULTS

Analysis and solved problem results shown that dimensional expansion of lower bound increases solution times and deviation results. But the gain of this priority algorithm is that the deviation amounts decreases with dimensional increasing of problem size and reaches a constant amount. Besides the behavior of priority algorithm against problems with various sizes is steady and predictable so that the operational quality and accuracy of this priority algorithm, which is very important, can be approximately predicted. This algorithm is also able to give optimized result to problems with $(6!)^6$ dimensional space and less than 3 to 5 percent errors. But it must be mentioned that the solving time of problems in this algorithm varies exponentially with the size of problems.

It seems if 6 and 9 equations are combined with the latest delivery time a new algorithm will be created which is able to optimize the least amount of total time and the most delay time in multiprocessor and semi processor job shop scheduling simultaneously. Moreover more researching on the suggested algorithm performance in order to reducing solving time and developing suggested algorithm in for job shop dynamic scheduling with regard to multiple time periods are suggested for more studding.

REFERENCES

[1] P.Brucker, J.Hurink and F. Warner, “Improving local search heuristic for some scheduling problems part II,” Discrete Applied Mathematics 72/1-2, 1997, 47-69.

[2] Blackstone, Philips and Hogg, "A state of the art survey of dispatching rules for manufacturing job-shop operations," International Journal of Production Research, Vol.20, 1982, pp.27-45.

[3] Biegel and Wink," Expert system can do job-shop scheduling: an exploration and a proposal," Computers and Industrial Engineering Vol.17/1-4, 1989, pp.347-352.

[4] Colorni, Dorigo, Maniezzo and Trubian,"Ant-system for job-shop scheduling," Belgian Journal of Operations Research, Statics and Computer Science,Vol. 34/1,2002, pp. 39-54.

[5] G.Shi," A genetic algorithm applied to a classic job-shop scheduling problem, "International Journal of Systems Science, Vol. 28/1, 1997, pp.25-32.

[6] P. Brucker,B.Jurisch,B.Sievers," A branch and bound algorithm for the job-shop scheduling problem," Discrete Applied Mathematics, Vol. 49,1994, pp. 109-127.

[7] E.Demirkol, S.Mehta, R.Uzsoy," A computational study of shifting bottleneck procedures for shop scheduling problems," Journal of Heuristic, Winter, Vol. 3/2, 2003, pp. 111-137.

[8] E.Nowicki , C.Smutnicki, "A fats taboo search algorithm for the job-shop problem," Management Science, Vol. 42/6, 1996, pp. 797-813.

[9] M.Kolonko," Some new results on simulated annealing applied to the job-shop scheduling problem," European Journal of Operational Research , Vol. 21,1998, pp. 112-121.

[10] A.S.Jain, S.Meeran," Job-shop scheduling using neural networks, "International Journal of production research , Vol. 36/5, 1998, pp. 1249-1272.

[11] E. Balas, "Machine scheduling via disjunctive graphs: An implicit enumeration algorithm," European Journal of Operational Research , Vol. 17,1969, pp.941-957.

[12] T.Yamada,R.Nakano, "Job-shop scheduling by simulated annealing combined with deterministic local search, "MIC’95 Metaheuristic International Conference, Hilton, Breckenridge, Colorado, USA ,1995, pp.344-349.

[13] B.Grabot,L.Geneste, "Dispatching rules in scheduling: A fuzzy approach, "International Journal of production Research,Vol. 32/4, 1994, pp. 903-915.

[14] T.Yamada,R.Nakano," Scheduling by genetic local search with multi-step crossover, PPSN’IV" Fourth International Conference on parallel problem solving from nature, Berlin, Germany, 1996, pp. 22-26.

[15] G.N.Varela , M.C.Sinclair, "Ant colony optimization for virtual –wave length-path routing and wave length allocation," In CEC99, Proceedings of the Congress on Evolutionary Computation, 1999, pp. 1809-1816.