Developing Multistage Heuristic Algorithm to Minimize Idle Time and Make Span in Job Shop Scheduling

Dr. Mahmoud Abbas Mahmoud*

Received on:18 /3/2009
Accepted on:6/5/2010

Abstract

The behavior and performance of job shops have been the focus and attention in both operations research and operations management literature. Job shop scheduling has received this large amount of attention, because it has the potential to dramatically decrease costs and increase throughput, thereby, profits. Moreover, the increasing of product customization creates more job shop environment in manufacturing world. No doubt, a wide variety of approaches to the modeling and solution of job shop scheduling problems have been reported in the literature. But, the research in this area is continuous. In this paper, Multistage Heuristic Algorithm based on priority dispatching rules is developed. This algorithm has been implemented to solve three cases. Schedules generated have been compared with those obtained by means of the basic algorithm. As a result, Multistage Heuristic Algorithm shows the ability of minimizing: machines idle time, total time of machines and makespan.

Keywords: Multistage Heuristic Algorithm, Job shop scheduling, Priority dispatching rules, Make span, Machines idle time.

بناء خوارزمية توجيهية متعددة المراحل لتدنية الوقت العاطل ووقت الأنهاء الأكبر في جدولة الأنتاج الوظيفي

الخلاصة

لقد لقى موضوع طبيعة وأداء نظم الانتاج الوظيفي اهتماما ملحوظا في أدبيات موضوعي بحوث العمليات وادارة العمليات على السواء. و يعودالسبب في هذا الحجم الواسع من الاهتمام بجدولة العمليات في نظم الانتاج الوظيفي لما لهذا الموضوع من أثر كبير في خفض الكلفة وزيادة الأنجاز وبالتالي زيادة الربح. اضافة لما ورد فان الازدياد بالتوجه نحو الانتاج التخصصي قد ادى الى توسع مساحة بيئة الانتاج الوظيفي في عالم التصنيع. لاشك بأنه يوجد في الأدبيات تنوعا واسعا في اساليب النمذجة والحل لمشكلة الجدولة هذه ولكن البحث في هذا المجال مازال مستمرا. في هذه الورقة تم بناء خوارزمية توجيهية متعددة المراحل مبنية على قواعد الاسبقيات في التقديم. هذه الخوارزمية قد طبقت في حل ثلاث حالات وتم مقارنة الجدولة الناتجة لكل حالة مع تلك الناتجة من تطبيق الخوارزمية الاساسية. وقد اوضحت النتائج بان الخوارزمية الجديدة ادت الى تقليل الوقت العاطل والوقت الكلي للمكائن وكذلك وقت الأنهاء الأكبر.

* Production Engineering and Metallurgy Department, University of Technology/ Baghdad

https://doi.org/10.30684/etj.28.14.8
2412-0758/University of Technology-Iraq, Baghdad, Iraq

.

1. Introduction

When we call a factory a "job shop", it means that this factory processes several jobs using shared recourses and the flow of raw and unfinished goods through it is completely random [1]. The increasing of product customization creates more job shop environment in manufacturing world. Over the years, the behavior and performance of job shops have been the focus and attention in literature. Recently, research papers on topics such as factory layout, inventory control, process control, production scheduling, and resource utilization can be found in almost every issue of every related journal. The most popular of these topics is production scheduling. It is often referred to as job shop scheduling. Obviously, job shop scheduling has received this large amount of attention, due to its potential to dramatically decrease costs and increase throughput, increase machines utilization, and increase profits as a result.

No doubt, a large number of approaches to the modeling and solution of these job shop scheduling problems have been reported in the literature, with varying degrees of success [1]. The main problem is to schedule the jobs so that the makespan, the time when all jobs have been completed, is minimized [2]. Obviously, various other objectives such as; minimizing total tardiness, minimizing total completion time, minimizing total flow time, etc., can be considered [3].

Generally, in job-shop we have a set of jobs that must be processed on a set of machines. A job consists of a sequence of operations, each of which is to be processed on a specific machine for a specified integral amount of time. Any job can have more than one operation on a given machine. The operations of a job must be processed in the given sequence, and a machine can process at most one operation at any given time. Therefore, routings in job shop environment is very complicated.

Practically, the numerous variables and constraints involved in job shop scheduling, complexity of its large solution space and its multi-criteria objective function make the problem difficult. The complexity of the problem can be seen from the fact that when n jobs go through m machines there are $(n!)^m$ possible schedules, hence if (n=20 and m=10) then the number of schedules is 7.2651×10^{183}[4]. This problem is a class of NP-Hard (Nondeterministic Polynomial time) ones that cannot be optimally solved for large-scale problems in a reasonable amount of computational time [5]. 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: priority dispatching rules [6], branch and bound method [7], local searching algorithm [5], dynamic programming [8], expert systems [9], genetic algorithms [10], simulation models [11], rolling horizon procedure [10], simulated annealing [13], tabu search, fuzzy logic, and neural network [1]. In addition to the above methods, hybrid methods can be also proposed such as, local search and simulated annealing algorithms [14] and fuzzy and priority dispatching rules algorithms [15]. No doubt, many other methods have been reported in literature.

2. Types of Schedules

Practically, for any job shop problem, there is infinite number of possible feasible schedules, because one can insert arbitrary amount of idle time at any machine between adjacent pairs of operations. This type of schedule is called the total possible schedules [16]. When the start time of a particular operation is constrained either by processing a different job on the same machine or by processing the directly preceding operation on different machines, this called semi-active schedule. The set of active schedules dominates the set of semi-active schedules in terms of optimizing any regular measure of performance. In the case, no machine is kept idle at a time when it could begin processing some operation then it is called a non-delay schedule [17].
The different types of schedules and relative sizes between them are illustrated in the Venn diagram shown in *figure (1)*.

3. Heuristic Dispatching Rules

Although, a large number of approaches to the modeling and solution of job shop scheduling problems have been reported in the literature, but practically heuristic dispatching rules has been extensively applied to the scheduling problems in job shop manufacturing comparing with the other approaches. These rules have strong advantages in that these are easy to understand, easy to apply, and require relatively little computing time. But, the primary disadvantage is that these cannot hope for an optimal solution [9].
Obviously, the terms dispatching rules, scheduling rules, sequencing rules, or heuristics are often used synonymously in literature [19]. There are enormous heuristic dispatching rules that have appeared in literature and practice, each can be used in scheduling jobs [20] [21]. The most popular rules are listed and described in *table (1)*.
No doubt, there are many other rules in practice. The most well known and comprehensive survey of scheduling heuristics is carried out by Panwalker and Iskander [19], where 113 rules are presented, reviewed and classified.

4. The Basic Algorithm for Job Shop Scheduling

Practically, in all above heuristic rules the basic algorithm for dispatching is as below:
Assume that machine m^* become idle, and denote J^* as set of jobs in the system that require to use of m^*. J^* may contain jobs that are waiting in the line at m^*. Jobs elsewhere in the system in line or being processed that eventually use m^*, and jobs that will soon be released to the jobs that use m^*.
Askin and Goldberg [21] listed the following three steps for sequencing jobs on the machines:

.

Step 1: Compute the set J^* of jobs that will be under consideration for processing next on m^*.

Step 2: For each job $j \in J^*$, compute a priority index (according to the rules that mentioned in the previous section).

Step 3: The next job on J^* is the job with best priority index value. If that job is immediately available, than start processing, otherwise keep m^* idle until the job arrives.

In the other hand, Panneerselvam [16] recommended five steps to generate active or non-delay schedule. These steps are as follows:

Let.

P_i be a partial schedule containing i scheduled operations

S_i the set of schedulable operations at stage i, corresponding to a given P_i

p_j the earliest time at which operation $j \in S_i$ could be started, and

q_j the earliest time at which operation $j \in S_i$ could be completed

p_j is determined by the completion time of the direct predecessor of operation j and the latest completion time on the machine required by operation j. The larger of these two quantities is p_j. The potential finish time q_j is simply $p_j + t_j$, where t_j is the processing time of operation j.

Hence, for a given priority rule R, a heuristic schedule generation is as in the following five steps:

Step 1: Let $t = 0$ and assume $p_t = \{\emptyset\}$.

S_t = {all operations with no predecessors}.

Step 2: determine $q^* = \min_{j \in S_i} \{ q_j\}$ and the corresponding machine m^* on which q^* could be realized.

Step 3: For each operation which belongs to S_t that requires machine m^* and satisfies the condition $p_j < q^*$ identify an operation according to a specific priority and add this operation to p_t as early as possible, thus creating only one partial schedule, p_{t+1} for the next stage.

Step 4: For each new partial schedule, p_{t+1} created in step 3, update the data set as follows:

a. Remove operation j from S_t.
b. Form S_{t+1} by adding the direct successor operation j to S_t.
c. Increment t by one.

Step 5: repeat from step 2 to step 4 for each p_{t+1} created in step 3 and continue in this manner until all active schedules are generated.

Similarly, another heuristic can be devised for the non-delay schedule generation by replacing $p_j < q^*$ with $p_j = q^*$ in step 3.

5. The Developed Algorithm

In practice, whenever there is a conflict (tie) in selecting an operation among competing operations, we will have to use a new priority rule as (tie breaker). Sometimes, priority rules are needed as tie breakers in more than one level. Neglecting resolving such conflicts is the main drawback of Askin and Goldberg algorithm.

.

Furthermore, the idea of keeping a machine idle while waiting for job, which is followed in both algorithms, may seem a bit odd.

Practically, both predecessor algorithms may lead to make one machine or more idle. Hence, a loss in machines utilization will be resulted. These machines can be loaded with other jobs by adjusting the start time of some operations towards left without affecting the sequence of operations to fill the non loaded times or in other words, filling the spaces on the Gantt chart.

In order to treat these weaknesses, the researcher developed a new algorithm. This algorithm is called Multistage Heuristic Algorithm. It aims to minimize makespan, and machines idle time. As a result it will maximize machines utilization. Moreover, this algorithm aims to resolve ties in any job shop problem.

The developed algorithm is based on a set of assumptions similar to that of the previous two algorithms but it uses three priority rules (R_1 , R_2, and R_3). The first is the basic priority rule while the other two as tie breakers, to resolve any conflict in selecting an operation among competing operations whenever it exist.

Scheduling by Multistage Heuristic Algorithm passes through the following steps:

Step 1: Enter data which contain number of jobs *j*, number of machines *m*, total number of operations to be scheduled *n*, process time for each operation t_{ij}, and operations sequence of each job.

Step 2: For each job, select the first operation depending on operations sequence.

Step 3: Group and list the first operations with their corresponding machines.

Step 4: For each machine, if there is only one operation waiting in front of it then schedule this operation on this machine. When there are more than one operation in front of machine, apply MWR as the first priority rule R_1 to select one operation as the winner among these compete operations. If R_1 does not resolve the conflict uniquely then apply SPT as the second priority rule R_2 to select one among the new compete operations. Similarly, if R_2 does not resolve the conflict uniquely then use the third priority rule R_3 to select one among the new compete operations. Since the third priority rule is the last one, so it must resolve the conflict. Hence, it is preferred to use "Random Rule" to pick any operation in the queue with equal probability.

Step5: Schedule all winner operations of the group of first operations by representing each one in the nearest possible space in the left side of the Gantt chart. Take in consideration that this space must equal to or excess the processing time of scheduled operation and this schedule is not affecting the sequence of operations.

Step 6: Omit the scheduled operations from the set of operations.

.

Step 7: For all none scheduled operations repeat the steps from *step 2* to *step 6* until all operations are scheduled.

The flow chart in *Figure (2)* illustrates the logic of Multistage Heuristic Algorithm.

6. Cases for Implementation and Comparison

This section is devoted to present three cases. The problems of these cases are solved first by means of the basic algorithm. For comparison, these problems are solved by implementing the developed Multistage Heuristic Algorithm next. All schedules which are obtained by means of these algorithms are represented in a form of Gantt charts in order to make the comparison easy and clear.

Case 1

This case is an example given and solved in details in reference [21]. This solution is done by means of the basic algorithm for job shop scheduling and according to Askin and Goldberg`s three steps for sequencing jobs on machines where LWR rule is used as priority rule. This case and the obtained schedule are shown in *table (2)* and *figure (3)* respectively. In fact, any try to solve this example with the same algorithm and steps but with using MWR rule instead of LWR, will lead to a tie (conflict). Hence, the solution can never be continued. A solution try and the tie are shown in *table (3)*. Ties problem has been explained in the predecessor paragraph.

Case 2

This case is an example given in reference [16] and shown in*table (4)*. The solution of this example is given in the same reference and carried out according to the steps that recommended by Panneerselvam for sequencing jobs on machines. The obtained schedule is shown in *figure (4)*.

Case 3

This case is an example given in reference [4] and shown in *table (5)*. This example has been solved by means of both Askin and Goldberg steps, and Panneerselvam`s steps in order to compare the obtained results together with that obtained by the developed Multistage Heuristic Algorithm.

The obtained schedule when Askin and Goldberg`s three steps are applied is shown in *figure (5)*. Note that LWR rule is used in solution. Similarly to case 1, any attempt to solve this example with the same steps but with MWR rule, will lead to a tie (conflict). The obtained schedule when Panneerselvam`s steps are applied in solving this example is shown in *figure (6)*.

Practically, the examples of the above three cases are solved by means of the developed Multistage Heuristic Algorithm. The obtained schedules for these examples are given as Gantt charts in *figure (7), figure (8), and figure (9)*. Moreover, the detailed steps of calculations are given in *table (6), table (7), and table (8).*

7. Results and Discussion

In order to have a detailed and clear discussion, the results of each case will be discussed one by one in the following paragraphs.

Case 1

From *figure (3)* and *figure (7)* it is clear that in this case, the total idle time of machines has been reduced from 30 in the original schedule (Askin and Goldberg steps) to 18 when the developed algorithm is applied. Hence, the total time of machines has been reduced from 130 to 118. As result, makespan also reduced from 54 to 44.

These results of this case and the details of each machine are summarized in *table (9).*

Case 2

Referring to *figure (4)* and *figure (8)* the total time of machines in this case has been reduced from 36 in the original schedule to 33 when the developed algorithm is applied while makespan is not reduced from its original value 13. In fact makespan not changed because machines 2 and 3 are fully loaded and there is no idle time to be reduced. But, the idle time of machine 1 has been reduced from 4 to 1.

The results and its details of this case are summarized in *table (10)*.

Case 3

From *figure (5), figure (6)* and *figure (9)* we can be notice that the total idle time in case 3 has been reduced from 19 in the schedule with Askin and Goldberg steps and 16 with Panneerselvam`s steps to 15 when the developed algorithm is applied. Hence, total time of machines has been reduced from 61 in the schedule with Askin and Goldberg steps and 58 with Panneerselvam`s steps to 57 when the developed algorithm is applied.

As result, Makespan reduced from 24 in the schedule with Askin and Goldberg steps and 22 with Panneerselvam`s steps to 21 when the developed algorithm is applied.

All results and its details of this case are given and summarized in *table (11)*.

8. Conclusions

Research in job shop scheduling area doesn't stop yet and still there is a room for improvement. A Multistage Heuristic Algorithm has been developed and presented in this paper. This algorithm aims to minimize machines idle time, makespan, and reducing machines time.

The performance of this algorithm has been tested through implementing it in solving three cases and comparing the results with those obtained when basic algorithm is applied. The result of comparison was positive and the test proofs the effectiveness of the Multistage Heuristic Algorithm since it shows ability to; reduce the total idle time of machines, reduce total time of machines, and reduced the makespan. Moreover, this algorithm shows an ability to resolve ties whenever exist in any job shop problem.

9. Future Work

Further research will focus on designing a computer aided system based on the developed multi stage heuristic algorithm, to provide an ease, fast and accurate tool to help users in solving job shop scheduling problems, especially when these problems are large.

References

[1]-Jones A., and Rabelo L. C., "Survey of Job Shop Scheduling Techniques", National Institute of Standards and Technology, Gaithersburg, MD, (1998).

[2]-Goldberg L. A., Paterson M., Srinivasan A., and Sweedyk E.

.

"Better Approximation Guarantees for Job Shop Scheduling" SIAM J. Discrete Math. Society for Industrial and Applied Mathematics Vol. 14 (1), (2001).

[3]-Applegate D. and Cook W., "A Computational Study of Job Shop Scheduling" ORSA Journal on Computing, 3 (1991).

[4]-Anant S. Jain, Sheik Meeran "Scheduling A Job-Shop Using A Modified Back-Error Propagation Neural Network" Proceedings of the IMS'96 Turkey, (1996).

[5]-Mahdavinejad R., "A New Approach to Job Shop-Scheduling Problem" International Journal of Quality Research, Vol. 1(1), (2007).

[6]-Vepsaiainen P. J., and Morton T. E., "Priority Rules for Job Shops with Weighted Tardiness Costs" Management Science, Vol. 33 (1987).

[7]-Artigues, C., Feillet, D "A branch and bound method for the job-shop problem with sequence-dependent setup times" Annals of Operations Research Vol.159 (1), (2008).

[8]-T. Lorigeon, J.C Billaut and J.L Bouquard "A Dynamic Programming Algorithm for Scheduling Jobs in a Two-Machine Open Shop with an Availability Constraint" *The Journal of the Operational Research Society*, Vol. 53, (11), (2002).

[9]-Gupta A., Sivakumar A., "Job Shop Scheduling Techniques in Semiconductor Manufacturing" International Journal of Advanced Manufacturing Technology Vol. 27: (2004).

[10]-James C., Mehmet E., and Terence C., "Evolving Genetic Algorithm for Job Shop Scheduling Problems" Proceedings of ACDM (2000).

[11]-Tavakkoli-Moghaddam R., and Daneshmand-Mehr M., "A Computer Simulation Model for Job Shop Scheduling Problems Minimizing Makespan" Computers & Industrial Engineering Vol. 48 (2005).

[12]-Bing Wang and Qiaoyun Li, "Rolling Horizon Procedure for Large-Scale Job-shop Scheduling Problems" International Conference on Automation and Logistics, Proceedings of the IEEE (2007).

[13]-Jeffcoat D. and Bulfin R. "Simulated Annealing for Resource-Constrained Scheduling" Europian Journal of Operational Research, Vol. 70 (1993).

[14]-Yamada T., and Nakano R., "Job-Shop Scheduling by Simulated Annealing Combined with Deterministic Local Search" MIC'95 Metaheuristic International Conference, Hilton, Breckenridge, Colorado, USA, (1995).

[15]-Grabot B., and Geneste L., "Dispatching Rules in Scheduling: A Fuzzy Approach" International Journal of Production Research, Vol. 32/4, (1994).

[16]-Panneerselvam R., "Production and Operations Management" Second Edition (2006).

[17]-Pinedo, M. L., "Scheduling, Theory, Algorithms, and Systems" 3rd Edition, Springer (2008).

[18]-T'kindt V., and Billaut J., "Multicriteria Scheduling Theory, Models and Algorithms" 2nd Edition, Springer, (2006).

[19]-Panwalker S., and Iskander W., "A Survey of Scheduling Rules" Operations Research Vol. 25(1), (1977).

[20]-Vollmann, T.E., Berry, W.L., and Whybark, D.C., "Manufacturing

.

Planning and Control Systems" Richard D. Irwin Inc., (1988).
[21]-Askin, R. G, and Goldberg, J. B. "Design and Analysis of Lean Production Systems" John Wiley & Sons, Inc. USA, (2002).

.

Table (1) Most popular heuristic dispatching rules and their description

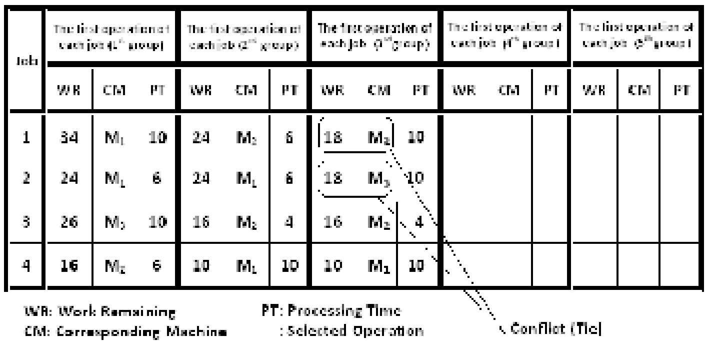
Rule	Rule abbreviation	Rule properties
Random	**R**	Pick any job in the queue with equal probabilit
First come first serve	**FCFS**	Jobs are processed in the order in which they arrived at the work center
Shortest processing time	**SPT**	This rule tends to reduce work-in-process inventory, the average job completion time, and average job lateness
Longest processing time	**LPT**	This rule tends to move the jobs with longest processing time in the work centers as soon as possible
Earliest due date	**EDD**	This rule seems to work well for critera associated with job lateness
Critical ratio	**CR**	In this rule the priority index is calculated by dividing the tim remaining until a job`s due date by the total shop time remaining for the job
Least work remaining	**LWR**	This rule is an extension of SPT in that it considers all of the processing time remaining until the job is completed. It tends to get the small jobs out of the shop quickly
Most work remaining	**MWR**	This rule is an extension of LPT in that it considers all of th processing time remaining until the job is completed. It tends to get the long jobs out of the shop quickly
The critical path	**CP**	This rule, always selects as the next job the one that is at head of the chain of jobs that contains the largest amount of processing
Slack time per operation	**ST/O**	A variant of ST that divides the slack time by the number o remaining operations, sequencing jobs in order of the smallest value first

Table (2) The example of case 1 [21]

Job	Routing - machine number, processing time require(			
1	M_1, 10	M_2, 6	M_3, 10	M_2, 8
2	M_1, 6	M_3, 10	M_2, 8	
3	M_3, 10	M_2, 4	M_1, 4	M_3, 8
4	M_2, 6	M_1, 10		

.

Table (3) The tie in solution trial for case 1 with Askin and Goldberg steps and MWR rule



Job	The first operation of each job (1st group)			The first operation of each job (2nd group)			The first operation of each job (3rd group)			The first operation of each job (4th group)			The first operation of each job (5th group)		
	WR	CM	PT	WR	CM	PT	WR	CM	PT	WR	CM	PT	WR	CM	PT
1	34	M_1	10	24	M_2	6	18	M_2	10						
2	24	M_1	6	24	M_1	6	18	M_3	10						
3	26	M_3	10	16	M_2	4	16	M_2	4						
4	16	M_2	6	10	M_1	10	10	M_1	10						

WR: Work Remaining
CM: Corresponding Machine
PT: Processing Time
: Selected Operation
Conflict (Tie)

Table (4) The example of case 2 [16]

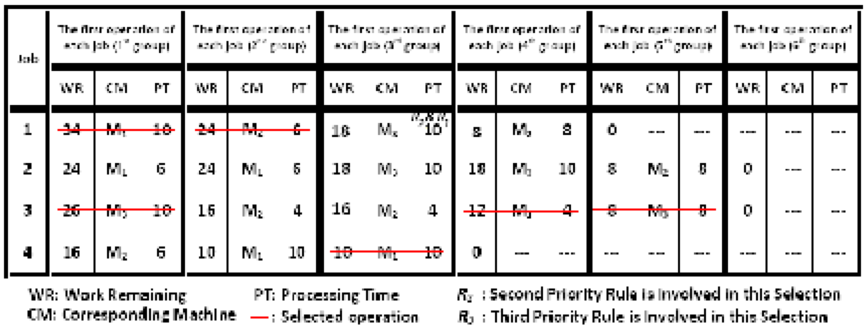
Job	Routing - machine number, processing time required		
1	M_1, 2	M_2, 3	M_3, 4
2	M_3, 4	M_2, 4	M_1, 1
3	M_2, 2	M_3, 2	M_1, 3
4	M_1, 3	M_3, 3	M_2, 1

Table (5) The example of case 3 [4]

Job	Routing - machine number, processing time required		
1	M_1, 1	M_2, 5	M_3, 3
2	M_2, 6	M_3, 3	M_1, 4
3	M_3, 5	M_1, 2	M_2, 1
4	M_2, 5	M_1, 4	M_3, 3

.

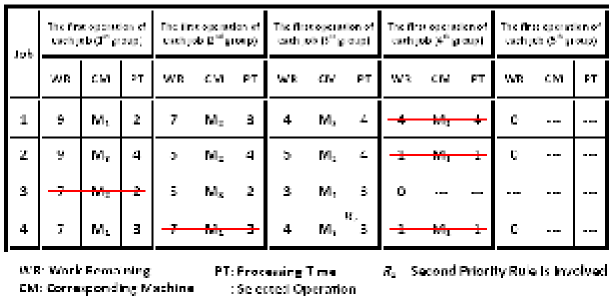
Table (6) The detailed calculations for case 1 by means of the developed algorithm



Job	The first operation of each job (1st group)			The first operation of each job (2nd group)			The first operation of each job (3rd group)			The first operation of each job (4th group)			The first operation of each job (5th group)			The first operation of each job (6th group)		
	WR	CM	PT	WR	CM	PT	WR	CM	PT	WR	CM	PT	WR	CM	PT	WR	CM	PT
1	~~34~~	~~M_1~~	~~10~~	~~24~~	~~M_2~~	~~6~~	18	M_4	R_2 & R_3 10	8	M_3	8	0	---	---	---	---	---
2	24	M_1	6	24	M_1	6	18	M_3	10	18	M_3	10	8	M_2	8	0	---	---
3	~~26~~	~~M_3~~	~~10~~	16	M_2	4	16	M_2	4	~~12~~	~~M_3~~	~~4~~	~~8~~	~~M_3~~	~~8~~	0	---	---
4	16	M_2	6	10	M_1	10	~~10~~	~~M_1~~	~~10~~	0	---	---	---	---	---	---	---	---

WR: Work Remaining PT: Processing Time R_2 : Second Priority Rule is Involved in this Selection
CM: Corresponding Machine — : Selected operation R_3 : Third Priority Rule is Involved in this Selection

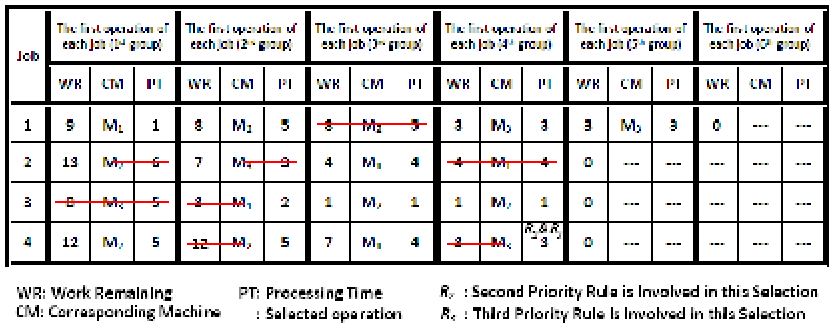
Table (7) The detailed calculations for case 2 by means of the developed algorithm



Job	The first operation of each job (1st group)			The first operation of each job (2nd group)			The first operation of each job (3rd group)			The first operation of each job (4th group)			The first operation of each job (5th group)		
	WR	CM	PT	WR	CM	PT	WR	CM	PT	WR	CM	PT	WR	CM	PT
1	9	M_1	2	7	M_2	3	4	M_1	4	~~4~~	~~M_3~~	~~4~~	0	---	---
2	9	M_1	4	5	M_2	4	5	M_1	4	~~1~~	~~M_1~~	~~1~~	0	---	---
3	~~7~~	~~M_2~~	~~2~~	5	M_2	2	3	M_1	3	0	---	---	---	---	---
4	7	M_1	3	~~7~~	~~M_1~~	~~3~~	4	M_1	R_2 3	~~1~~	~~M_3~~	~~1~~	0	---	---

WR: Work Remaining PT: Processing Time R_2 Second Priority Rule is Involved
CM: Corresponding Machine : Selected Operation

Table (8) The detailed calculations for case 3 by means of the developed algorithm



Job	The first operation of each job (1st group)			The first operation of each job (2nd group)			The first operation of each job (3rd group)			The first operation of each job (4th group)			The first operation of each job (5th group)			The first operation of each job (6th group)		
	WR	CM	PT	WR	CM	PT	WR	CM	PT	WR	CM	PT	WR	CM	PT	WR	CM	PT
1	9	M_1	1	8	M_3	5	~~8~~	~~M_2~~	~~5~~	3	M_3	3	3	M_3	3	0	---	---
2	13	~~M_3~~	~~6~~	7	~~M_4~~	~~9~~	4	M_1	4	~~4~~	~~M_1~~	~~4~~	0	---	---	---	---	---
3	~~8~~	~~M_2~~	~~5~~	~~3~~	~~M_1~~	2	1	M_2	1	1	M_2	1	0	---	---	---	---	---
4	12	M_3	5	~~12~~	~~M_3~~	5	7	M_1	4	~~3~~	~~M_4~~	R_2 & R_3 3	0	---	---	---	---	---

WR: Work Remaining PT: Processing Time R_2 : Second Priority Rule is Involved in this Selection
CM: Corresponding Machine : Selected operation R_3 : Third Priority Rule is Involved in this Selection

.

Table (9) Summery of the results of case 1

Machines	Total idle time		Machine time		Makespan	
	Askin and Goldberg steps	Developed algorithm	Askin and Goldberg steps	Developed algorithm	Askin and Goldberg steps	Developed algorithm
M_1	0	0	30	30	54	44
M_2	22	12	54	44		
M_3	8	6	46	44		
Total	30	18	130	118		

Table (10) Summery of the results of case 2

Machines	Total idle time		Machine time		Makespan	
	Panneerselvam`s steps	Developed algorithm	Panneerselvam`s steps	Developed algorithm	Panneerselvam`s steps	Developed algorithm
M_1	4	1	13	10	13	13
M_2	0	0	10	10		
M_3	0	0	13	13		
Total	4	1	36	33		

Table (11) Summery of the results of case 3

Machines	Total idle time			Machine time			Makespan		
	Askin and Goldberg	Panneerselvam	Developed	Askin and Goldber	Panneerselvam	Developed	Askin and Goldberg	Panneerselvam	Developed
M_1	13	8	8	24	19	19	24	22	21
M_2	0	0	0	17	17	17			
M_3	6	8	7	20	22	21			
Total	19	16	15	61	58	57			

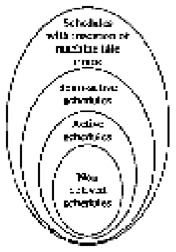


Figure (1) Venn diagram showing different types of schedules [18]

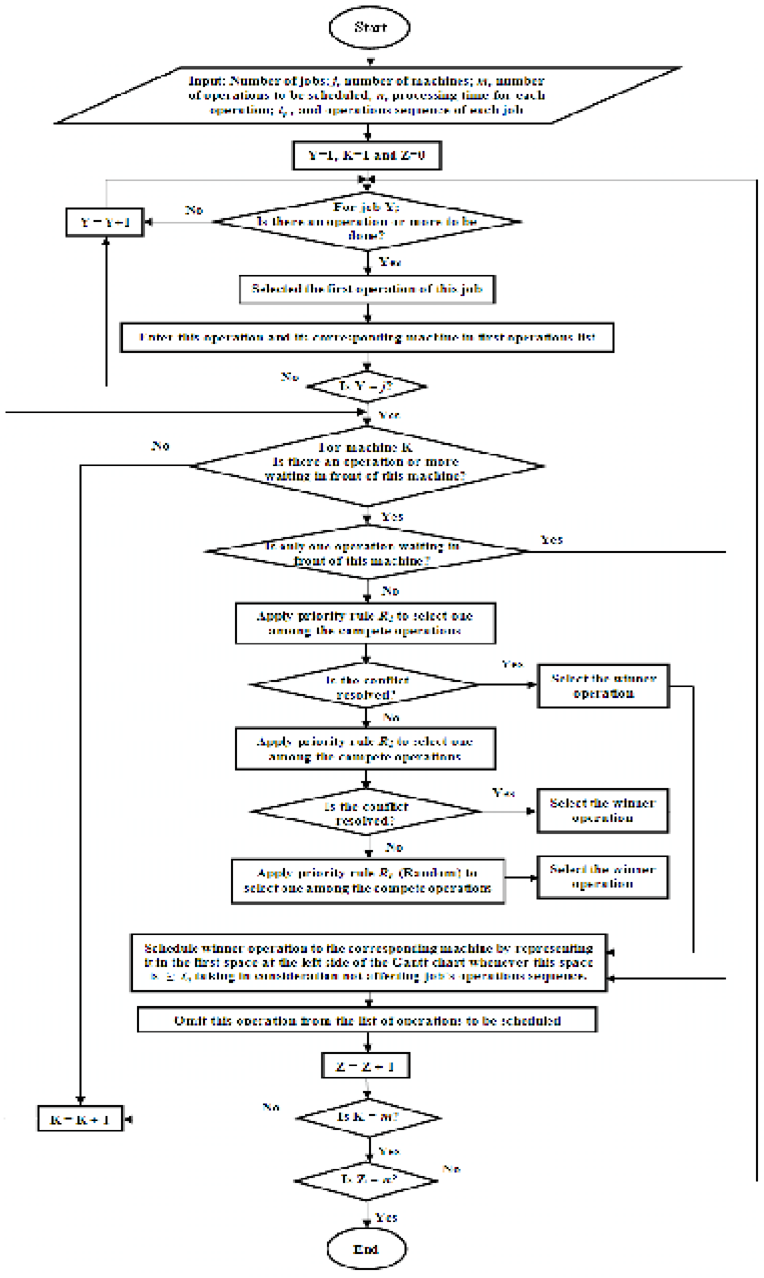
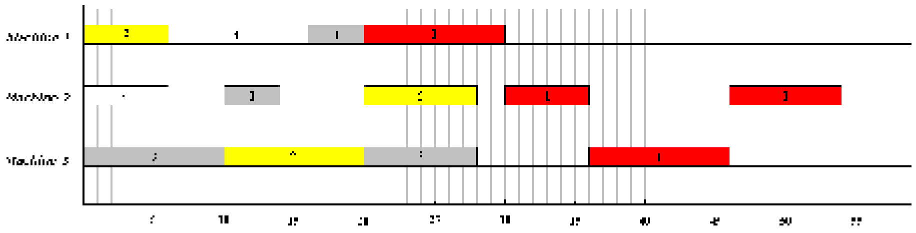
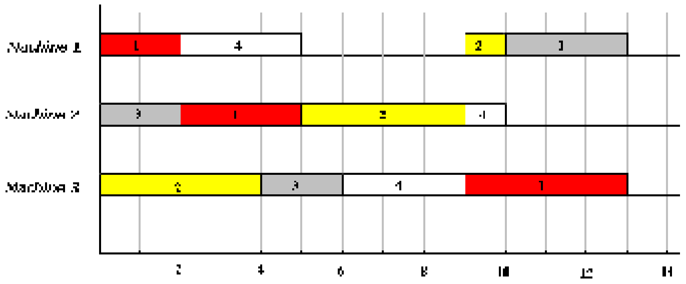

Start
Input: Number of jobs; j, number of machines; m, number of operations to be scheduled, n, processing time for each operation; t_{ij}, and operations sequence of each job
Y=1, K=1 and Z=0
For job Y; Is there an operation or more to be done?
No
Y = Y+1
Yes
Selected the first operation of this job
Enter this operation and its corresponding machine in first operations list
No
Is Y = j?
Yes
No
For machine K Is there an operation or more waiting in front of this machine?
Yes
Is only one operation waiting in front of this machine?
Yes
No
Apply priority rule R_1 to select one among the compete operations
Is the conflict resolved?
Yes
Select the winner operation
No
Apply priority rule R_2 to select one among the compete operations
Is the conflict resolved?
Yes
Select the winner operation
No
Apply priority rule R_3 (Random) to select one among the compete operations
Select the winner operation
Schedule winner operation to the corresponding machine by representing it in the first space at the left side of the Gantt chart whenever this space is ≥ t_{ij} taking in consideration not affecting job's operations sequence.
Omit this operation from the list of operations to be scheduled
Z = Z + 1
No
K = K + 1
Is K = m?
Yes
Is Z = n?
No
Yes
End

Figure (2) The logic of Multistage Heuristic Algorithm



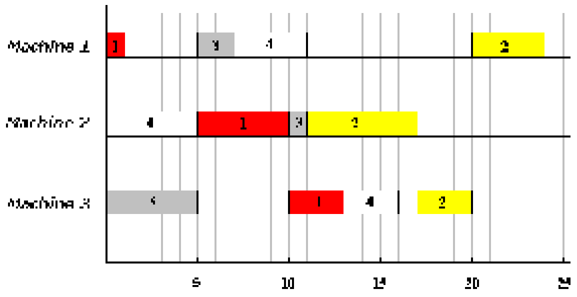
Total Time of Machines = 130
Makespan = 54
Machines Idle Time = 30

Figure (3) The schedule obtained by Askin and Goldberg for case 1



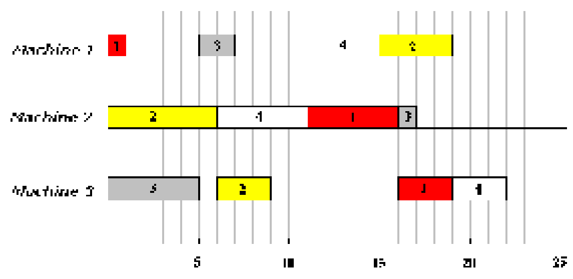
Total Time of Machines = 36
Makespan = 13
Machines Idle Time = 4

Figure (4) The schedule obtained by Panneerselvam for case 2



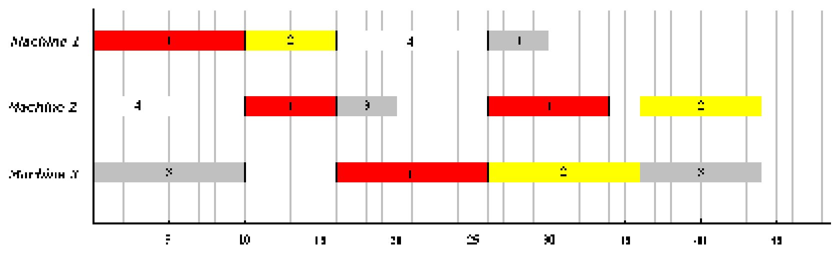
Total Time of Machines = 61
Makespan = 24
Machines Idle Time = 19

Figure (5) The schedule obtained with Askin and Goldberg steps for case 3



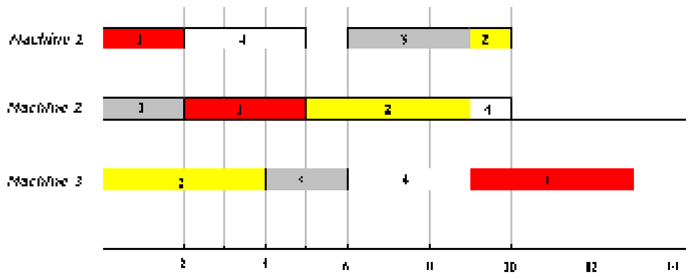
Total Time of Machines = 58
Makespan = 22
Machines Idle Time = 16

Figure (6) The schedule obtained with Panneerselvam`s steps for case 3



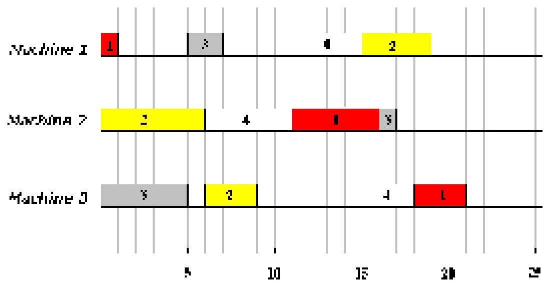
Total Time of Machines = 118
Makespan = 44
Machines Idle Time = 18

Figure (7) The obtained schedule for case 1 by means of the developed algorithm



Total Time of Machines 33
Makespan 13
Machines Idle Time 1

Figure (8) The obtained schedule for case 2 by means of the developed algorithm



Total Time of Machines = 57
Makespan = 21
Machines Idle Time = 15

Figure (9) The obtained schedule for case 3 by means of the developed algorithm