Advances in Mathematics: Scientific Journal 8 (2019), no.3, 35–42
(Special issue on ICRAPAM)
ISSN 1857-8365 printed version
ISSN 1857-8438 electronic version

AN HEURISTIC APPROACH FOR SOLVING OSSP WITH A OBJECTIVE OF MINIMIZING TOTAL COMPLETION TIME AND RESOURCE IDLENESS

Dr. S. Jayakumar[1], M. Sophia[2] and A. Dharani[3]

[1&2]*Assistant Professor & Head, PG & Research Department of Mathematics, AAGAC, Cheyyar.*

[3]*Research Scholars, PG & Research Department of Mathematics, AAGAC, Cheyyar, TN, India.*

ABSTRACT: This article mainly focused on open shop scheduling with the objective of minimizing the total completion time and resource idleness. Many researchers assume that the total completion time alone reduces the resource idleness. Since for more than two jobs two machine case the problem is NP – Hard in nature due to the complexity of the problem. Also note that the order of processing is any conceivable order it further complicate the problem. Literature review reavels that only for two job two machine cases available upto 2000. In the year 2001 S.Jayakumar solved the open shop scheduling problem with objective of minimizing the makespan and resource idleness for the general n job m machine cases. Later, during the year 2017 Jayakumar and Meganathan solved the same problem with release dates and it was found the no of jobs greater than no of machines and no of jobs equal to no of machines the proposed LPT based algorithm performs better. In this paper also the LPT based algorithm performs better than the other two.

1. Introduction

Scheduling is nothing but allocation of resources over a period of time in order to perform a collection of tasks. Scheduling can be broadly classified into three categories, namely the flow shop scheduling problem, Job shop scheduling problem and open shop scheduling problem. If the jobs are purely sequencing it is called flow shop scheduling problem. If the jobs are process through sequencing and routing it is called job shop scheduling problem and in open shop scheduling problem jobs can be processed in any conceivable order.

During the year 1976 Gonzales and Sahni introduced the open shop scheduling problem with an objective is to minimize finish time. They have solved the problem for upto 2 job 2 machine problem only for more than 2 job 2 machine case literature was not found till 2001. In the year 2001 Jayakumar solved this problem using a heuristic approach to solve not only for the 2 job 2 machine case but also n job m machine case in general with objective of minimizing the makespan and resource idleness. He developed three algorithms and compared with these three algorithms and found which algorithm performs better than the other two. In the year 2017, Meganathan solved the open shop scheduling problem with the objective of minimizing the makespan and resource idleness for release dates. In that Thesis he concluded that the DLPT-DS performs better than the DSPT-DS algorithm.

In the year 2019 we have made our investigation intoopen shop scheduling problems with the objective of minimizing the total completion time and resource idleness. The algorithm developed by Jayakumar (2001) is chosen in order to check, whether the makespan minimization algorithms also performing in minimizing the total completion time and resource idleness. It was found that the LPT based algorithm performs better than the other two.

Sequencing and scheduling

Scheduling is the allocation of resources over time to perform a collection of tasks and it is a decision making function. The practical problem of allocating resources over time to perform a collection of tasks arises in a variety of situations. In most cases, however, scheduling does not become a concern until some fundamental planning problems are resolved, and it must be recognized that scheduling decisions are of secondary importance to a broader set of managerial decisions. The scheduling process must often arise in a situation where resource availability fixed by the long-term commitments of a prior – planning horizon.

Sequencing is the order of processing a set of tasks over available resources. Scheduling involves sequencing task of allocating as well as the determination of process, commencement and completion times i.e., time-tabling. Sequencing problems occur whenever there is a choice to the order in which a group of tasks can be performed. The shop supervisor or scheduler can deal with sequencing problems in a variety of ways. The simplest approach is to ignore the problem and accomplish the talks in any random order. The most frequently used approach is to schedule heuristically according to predetermined “rules of thumb”. In certain cases. Scientifically derived scheduling procedures can be used to optimize the scheduling objectives.

Sequencing problem is the problem of finding an optimal sequence of completing a certain number of jobs so as to minimize the total elapsed time between completion of the first and last job.

Open shop scheduling problem:

In the event of having ‘n’ jobs available for processing through ‘m’ machines. Each job can be processed in any order. The processing of a given job on a given machine is called an operation. We shall denote the operation on the i^{th} job on the j^{th} machine by (i, j). The processing times are denoted by ‘t_{ij}’.

Each operation (i, j) takes a certain length of time, the processing time to perform. We denote this by ‘t_{ij}’ let us assume that the processing times are arbitrary chosen. The general problem is to find an optimal schedule on the basis of a certain measure of performance. In the area of open shop scheduling the objective is being decided using various factors like quality, promptness, customer satisfaction, minimizing makespan, utilization of the resources in optimally, etc.

An open shop problem is a special case of the general shop in which

- Each job i consists of m operations O_{ij} (j = 1,2, … m) where O_{ij} must be processed on machine M_j,
- There are no precedence relations between the operations.
- Thus the problem is to find job orders and machine orders.

Notations: terminology and assumptions

Notations:

t_{ij}	-	Processing time (time required) for job i on machine j.
m	-	Number of machines
n	-	Number of Jobs
C_i	-	Completion time of the last operation in the assumed sequence for processing of the job i.
Max	-	Maximum
LPT	-	Longest processing time
SPT	-	Shortest processing time
MWP	-	Maximum work pending
S_k	-	be the set of all operations involved in a problem
P_k	-	is the partial list of the schedule
P*	-	min(p_{ij}) be the earliest time at which operation (i,j) ϵ S_k be started.
TCT	-	Total Completion Time

Terminology:

- **Number of Machines**: The number of machines refer to the number of service facilities through which a job must pass before it is assumed to be completed.
- **Processing time**: This is the time required by a job on each machine
- **Completion time**: It is the time at which the job is completed in a sequence. Performance measures of evaluating schedules are usually fuction of job completion time. Some sample performance measures are flow time, Lateness, Tradiness.
- **Makespan**: Makespans is the maximum completion time among n scheduled jobs. Also it can be defined as completion time at which all jobs completed its processing.

Assumptions:

- There are n jobs ready for processing at time t=0
- Pre-emption not allowed each job once started in that machine
- Each job has 'm' distinct operations, one on each machine
- A machine also may be idling for want of jobs.
- The jobs can be processed in any order ie, there is no technological constraint.
- No machine may process more than one job at a time.
- The setup time and transportation time between the machines are included in the processing time, ie, the transfer time between machine is considered negligible compared with the processing time.
- There is no randomness. In particular,
  - The number of jobs are known and fixed
  - The number of machines are known and fixed
  - The processing time are known and fixed
- Machines to be used are of different types.

This process will be continued until all jobs are completed on each machine with given processing times.

Literature review of open shop scheduling problem

The scheduling problems are classified into single machine scheduling, flow shop scheduling, job shop scheduling open shop scheduling and hybrid scheduling. In this paper, the open shop scheduling problem is considered. The open shop scheduling problem is alternatively called as moderated job shop scheduling problem, which is between the flow shop scheduling problem and job shop scheduling problem.

The flow shop scheduling problem consists of "n" jobs, each with "m" operations. The process sequences of these jobs are one and the same for this problem. The open shop scheduling problem consists of n jobs, each with m operations. The process sequences of the jobs are not the same for this problem. The open shop scheduling problem consists of n jobs, each with atmost m operations.

If a problem consists of n jobs, each with atmost m operations of open shop scheduling problem, then a generalized data format of the processing times is shown in Table 1, if t_{ij} is positive, then the job i requires t_{ij} units of processing time for the operation j.

In open shop scheduling problem (OSSP) there is a finite set J which consists of n jobs $\{J_1, J_2 \ldots\ldots J_n\}$ and a set M which consists of m machines $\{M_1, M_2 \ldots\ldots\ldots M_n\}$. Each job J_i (i=1 to n) is to be processed on machine M_j (j = 1 to m) for t_{ij} processing time, where ij stands for i^{th} job on a j^{th} machine. Each job J_i consists of atmost m tasks. At a time, each job can be processed only on one machine and each machine can process only one job.

Measures of performance of open shop problem:
The scheduling of the jobs in the open shop scheduling has several measures of performance which are as listed below:

- ❖ Minimize the makespan;
- ❖ Minimize the Total Completion Time;
- ❖ Minimize the Total Completion Time and Resource Idleness.

Let

- ⇒ n be the number of jobs;
- ⇒ m be the number of machines (operations);
- ⇒ c_i be the completion time of the last operation in the assumed sequence for processing of the job i ;

Statement of the problem:
The objective is to find a sequence of jobs with the given processing times on each machine to minimize the total completion time and resource idleness of machines as well.

This sections presents review of literature on the minimization of the sum of completion time of all the jobs under the following subdivisions:

- ⇒ Exact algorithm;
- ⇒ Heuristics;
- ⇒ Tabu search algorithm;
- ⇒ Particle swarm optimization algraithm;
- ⇒ Hybrid algorithm;
- ⇒ Multiple algorithm;
- ⇒ Miscellaneous algorithm.
- ⇒

2. Heuristics

Achugbue and chin considered the open shop scheduling problem with the objective of minimizing the mean flow time. They showed that this problem is NP – complete. In addition, they derived bounds for mean flow times of arbitrary schedules and SPT first schedules for m-process and n-job systems in terms of the mean flow time of the optimal schedule. Werra and Blazewicz presented an edge colouring model for preemptive open shop problem, in which additional constraints generated by the presence of resource R are considered, where resource “R” can be nonrenewable resource (money, fuel, etc.) or a renewable resource (Manpower, tools, etc.) The objective of this problem is to minimize the total completion time. They provided edge colouring of bipartite graph for both problems, viz, preemptive with renewable Resource open shop (PROS) and preemptive with nonrenewable resource open shop (PNOS). They further provide basic properties of edge colourings.

The authors used the algorithm developed by him for Minimizing the Makespan and Resource Idleness which has been tested for the objective of Minimizing the Total Completion Time and

Resource Idleness. It has been found that the proposed LPT based algorithm performed better than the other two algorithms, with the objective of minimizing the Total Completion Time and Resource Idleness.

An heuristic algorithm using lpt:

MWP(Maximum Work Pending) : select the operation associated with the job having the most work pending to be processed.

LPT (Longest Processing Time) : select the operation with the maximum processing time.

RAN(Random) : If there is a tie, Select the operation at random. At each stage, it is necessary to identify the operation 'S_k'and to keep track of the times at which the machines are available for processing. MWP_{IJ} is the work pending of the job associated with operation (i, j).

Heuristic Schedule Generation:

Step 1: Let t=0 and assume P_k=(Empty), S_k =(All operation).

Step 2: Examine P^*= min (P_{ij}), (i, j) in S_k as explained and the corresponding operation for which P^* could be released. If 'P^*'occurs only for operation in S_K . Then add that operation to 'P_k' and create the next partial schedule P_{k+1} otherwise go to step 3.

Step 3: Among the operation in S_K for which 'P_{ij}' is equal to P^*, identify an operation according to the order of priorities as given in the earlier section and add this operation to 'P_k' as early as possible, thus creating only on partial schedule 'P_{k+1}' for the next stage.

Step 4: For each new partial schedule 'P_{k+1}' created in step 3, update the data set as follows: From 'S_{k+1}' by deleting operation (i, j) from S_K. Increment 't' by one.

Step 5: Repeat from step 2 to step 4 for each P_{k+1} created in step 3 and continue in this manner until all the operations are added in to 'S_K'.

An heuristic algorithm using spt:

Priority Rules:

MWP (Maximum Work Pending): select the operation associated with the job having the most work pending to be processed.

SPT (Shortest processing time): select the operation with the shortest processing time.

RAN (Random): If there is a tie, select the operation at random. At each stage, it is necessary to identify the operation in 'S_K' and to keep track of the times at which the machines are available for processing. $MWP_{ij}$is the work pending of the job associated with operation (i, j).

Heuristic schedule generation:

Step 1: let t=0 and assume P_k=(empty), S_k = {All operation}.

Step 2: Determine P^*= min (P_{ij}), (i, j) in S_K as explained and the corresponding operation for which P^* could be released. If 'P^*'occurs only for operation in S_K . Then add that operation to 'P_k' and create the next partial schedule P_{k+1}otherwise go to step 3.

Step 3: Among the operation in S_K for which 'P_{ij}' is equal to P^*,identify an operation according to the order of priorities as given in the earlier and add this operation to 'P_k' as early as possible, thus creating only on partial schedule 'P_{k+1}' for the next stage.

Step 4: For each new partial schedule 'P_{k+1}' created in step 3, update the data set as follows: From 'S_{k+1}' by deleting operation (i, j) from S_K. Increment 't' by one.

Step 5:P_{ij}= 0 from step 2 to step 4 for each P_{k+1} created in step 3 and finding in this manner until all the operations are added in to 'S_K'.

An heuristic algorithm using random

Priority Rules:

MWP (Maximum Work Pending): select the operation associated with the job having the most work pending to be processed.

RAN (Random): Select the operation at random. At each stage, it is necessary to identify the operation in 'S_K' and to keep track of the times at which the selections are available for processing. $MWP_{ij}$is the work pending of the job associated with operation (i, j).

Heuristic schedule generation:

Step 1: Let t=0 and assume $\boldsymbol{P_k}$=(empty), $\boldsymbol{S_k}$ = {All operation}.

Step 2: Determine $\boldsymbol{P^*}$ = min ($\boldsymbol{P_{ij}}$), (i, j) in $\boldsymbol{S_K}$ as explained and the corresponding operation for which $\boldsymbol{P^*}$ could be released. If '$\boldsymbol{P^*}$'occurs only for operation in S_K . Then add that operation to '$\boldsymbol{P_k}$' and create the next partial schedule $\boldsymbol{P_{k+1}}$ otherwise go to step 3.

Step 3: Among the operation in S_K for which '$\boldsymbol{P_{ij}}$' is equal to $\boldsymbol{P^*}$,identify an operation according to the order of priorities as given in the earlier and add this operation to '$\boldsymbol{P_k}$' as early as possible, thus creating only on partial schedule '$\boldsymbol{P_{k+1}}$' for the next stage.

Step 4: For each new partial schedule '$\boldsymbol{P_{k+1}}$' created in step 3, update the data set as follows: From '$\boldsymbol{S_{k+1}}$' by deleting operation (i, j) from $\boldsymbol{S_K}$. Increment't' by one.

Step 5: Repeat from step 2 to step 4 for each $\boldsymbol{P_{k+1}}$ created in step 3 and continue in this manner until all the operations are added in to '$\boldsymbol{S_K}$'.

Three algorithms have been developed to achieve the objective that of minimizing the total completion time. For more than two machines with arbitrary processing time, the investigation is made between the three algorithm presented above and it as been found that the algorithm 'A' performs better than the other two in finding total completion time. The computational result indicates that for the square matrix instances (number of jobs equal to the number of machines) as well as the rectangular matrix instances (the number of jobs greater than the numbers of machines) the proposed Algorithm 'A' performs better than the other two. Whereas the number of jobs less than the number of machine instances the proposed algorithm 'B' performs better than the other two.

3. Appendix

Illustration:

	M1	**M2**	**M3**
J1	**6**	**9**	**7**
J2	**5**	**7**	**9**
J3	**4**	**9**	**8**
J4	**6**	**9**	**5**

Solution:

An heuristic algorithm using LPT:

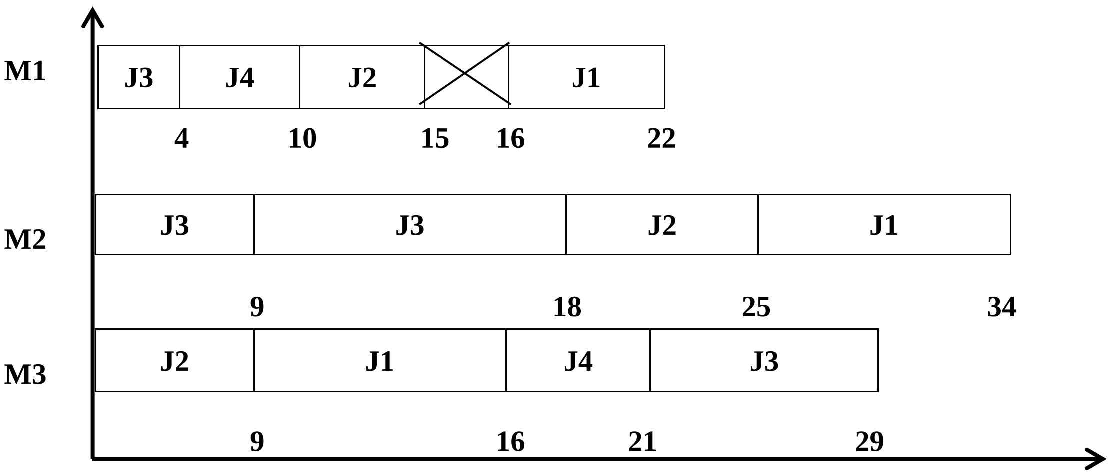

Fig 1: Gantt chart represents the allocation of jobs on machines using An heuristic algorithm based on LPT.

An heuristic algorithm using SPT:

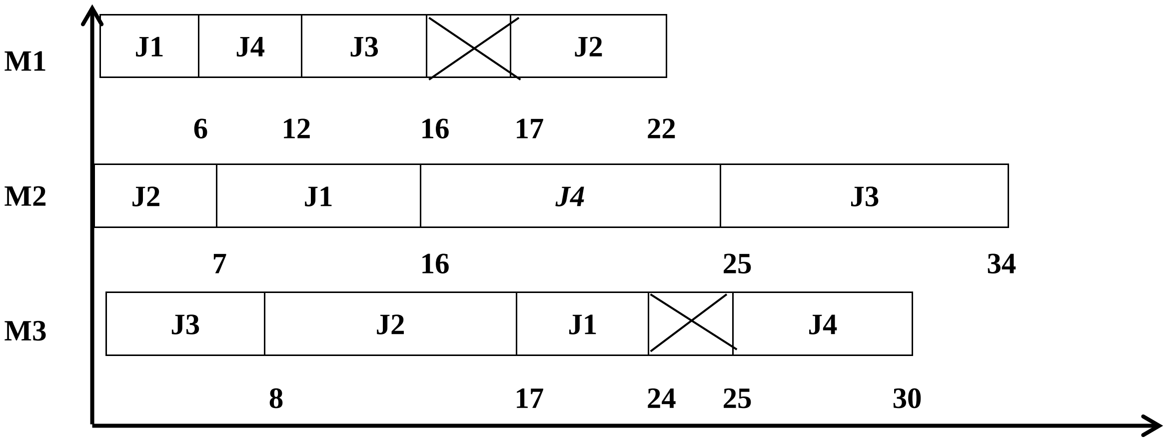

Fig 2: Gantt chart represents the allocation of jobs on machines using An heuristic algorithm based on SPT

An heuristic algorithm using Random:

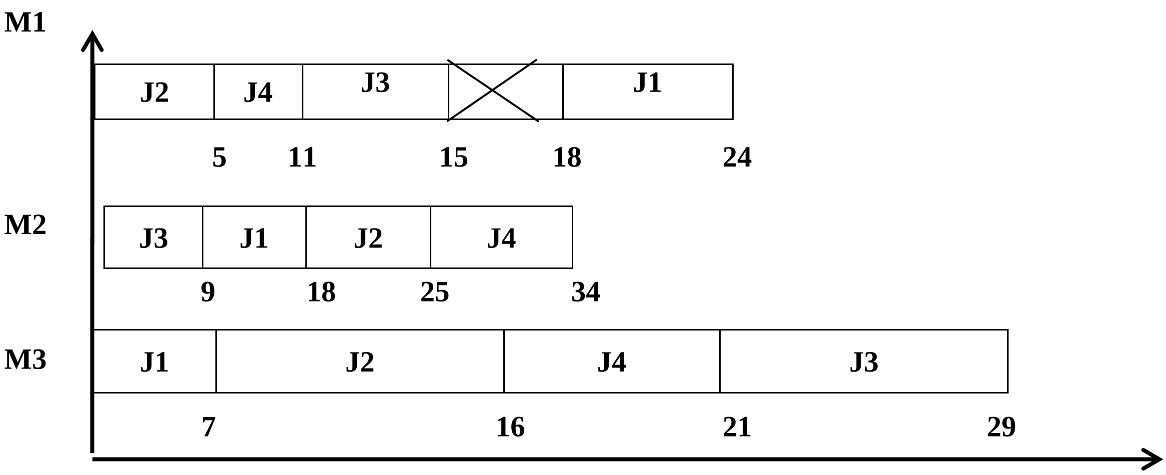

Fig 3: Ganttchart represents the allocation of jobs on machines using An heuristic algorithm based on Random

4. Results obtained

The total completion time and resource idleness for the heuristic algorithms are given as follows:

Sl.No	Algorithms	Total Completion Time	Resource Idleness
01	LPT	85 hrs	1hr
02	SPT	86 hrs	2hrs
03	RANDOM	87 hrs	3hrs.

5. Conclusion

As far as the open shop scheduling problem with the objective of minimizing the total completion time and resource idleness are concern the performance of the LPT based algorithm is superior than the other two algorithms for not only the 4 jobs 3 machine case but also for the 'n' job 'm' machine cases in general. As long as the processing time differs considerably the performance also varies.

References:

[1] Bai D and Tong L, 2013, “Open Shop Scheduling Problem to minimize makespan with release dates”, Applied mathematical modeling, 37, PP. 2008 - 2015.

[2] Baker K.R., (1974) “ Introduction to sequencing and scheduling”, Wiley, New York.

[3] Chen B, Potts C N and Woeginger G J, 1998, “ A review of machine scheduling, Complexity, algorithms and approximality”, in D-Z Du et al.

[4] Dorndorf U, Pesch E and Phan - Huy T, 2001, “Solving the open shop scheduling problem”, Journal of scheduling, 4, PP 157 -174.

[5] Ellur Anand[1], Ramasamy Panneerselvam[2] 2015 “Literature Review of open shop scheduling problems” PP. 33- 52.

[6] Gonzalez, T and Sahni, S(1976),“open shop scheduling to minimize Finish time”.

[7] Gerhard J. Woeginger, 2012, “The open shop scheduling problem” 4, PP. 4.1 - 4.12

[8] Jayakumar. S., (2001), “A Heuristic approach for solving open shop scheduling problem, Minimizing the makesapan and resource idleness”, Ph.D. Thesis, Anna University.

[9] Jayakumar S and Meganathan R, 2014, “ Heuristic approach for open scheduling problem to minimize with release dates ”, Industrial Engineering Journal, 7 (8), PP -17-22.

[10] Meganathan R, 2017, “Heuristic approach for solving OSSP with release dats, to minimize the makespan and Resource Idleness” Ph.D Thesis, Thiruvalluvar University, Vellore, Tamilnadu, India.

[11] B. Naderi[a,] S.M.T .FatemiGhomi[a,*], M. Aminnayeri[a,] M.Zandieh[b,] 2011 “Scheduling open shops with parallel machines to minimize total completion time,” PP:1275 - 1287.

[12] Thomas E. Mortan, and David W. Pentico, (1993), “Heuristic Scheduling Systems”, John Wiely and Sons, New York.

[13] C. WeiB[1]. S. Waldherr[2]. S. Knust[2.] N.V. Shakh le vich[1] 2016, "Open shop Scheduling Synchronization", 20, PP. 557 - 581.