Contents lists available online at TALENTA Publisher

DATA SCIENCE: JOURNAL OF COMPUTING AND APPLIED INFORMATICS (JoCAI)

Journal homepage: https://jocai.usu.ac.id

On Optimum Sequencing of Job Shop Scheduling in Manufacturing Shop

Nwozo C.R.[1] and Adewoye, S. O.[2*]

[1]*Department of Mathematics, University of Ibadan, Nigeria*
[2]*Department of Mathematics, Yaba College of Technology, Nigeria*

email: [1]*crnwozo@yahoo.com*, [2]*adewoye2012@gmail.com*

ARTICLE INFO

Article history:
Received 15 June 2023
Revised 24 June 2023
Accepted 21 July 2023
Published online 31 July 2023

Keywords:
Job shop scheduling problem
Optimization
Probability Sequencing

Corresponding Author:
adewoye2012@gmail.com

ABSTRACT

Joh shop scheduling problem (JSSP) is an NP Hard problem. The most obvious real-world application of the JSSP is within manufacturing and machining as the parameter description describes. Companies that are able to optimize their machining schedules are able to reduce production time and cost in order to maximize profits. The aim of the problem is to find the optimum schedule for allocating shared resources over time to complete all n jobs within the problem. In this research we employ probability sequencing and make of comparison of job-shop sequencing rule such as first-come, first-served (FCFS) rule, which can be accomplished only by used of digital simulation. The result shown that using probability sequencing, when the machine is free, a job is selected in accordance with a sequencing rule and is allowed to occupy the machine for a time equal to its predetermined processing time. A job is complete when it has been processed through all centers on its route.

IEEE style in citing this article:
C.R. Nwozo and S.O. Adewoye " On Optimum Sequencing of Job Shop Scheduling in Manufacturing Shop," DATA SCIENCE: JOURNAL OF COMPUTING AND APPLIED INFORMATICS (JoCAI), vol. 7, no. 2, pp. 104-110. 2023.

1. Introduction

Production sequencing decisions arise whenever there is a choice as to the order in which a number of jobs can be performed. The democratic and commonly accepted first-come, first-served rule may world well in society when people are waiting for service, but there are better sequencing rules when production orders or jobs are involved. The effectiveness of a specific sequence may be measured in terms of makespan time, average completion time, due date performance, machine utilization, inventory of jobs in process, and so forth.[7],[10]. Machine scheduling problems arise in diverse areas such as flexible manufacturing systems, production planning, computer design, logistics, communication, etc.

A scheduling problem is to find sequences of jobs on given machines with the objective of minimising some function of the job completion times. In a simpler version of this problem, *flow shopscheduling* [12], all jobs pass through all machines in the sameorder. A more complex case is

e- ISSN: 2580-829X | p- ISSN: 2580-6769

represented by a *job shop scheduling problem* where machine orderings can be differentfor each job. Job shop scheduling problem is one of the hardest combinatorial optimizationproblems. It belongs to the class of NP-hard problems,consequently there are no known algorithms guaranteed to give an optimal solution and run in polynomial time. That means, classical optimization methods (branch and bound method, dynamic programming) can be used only for small scale tasks. Therefore, more complex tasks must be solved by heuristic methods [15], [1]. Successful heuristic methodsinclude approaches based on *simulated annealing* [7], [12], and *genetic algorithms* [8], [17]. A veryefficient method combines a variable depth search procedurewith a *shifting bottleneck* framework [1], [19].

The papers[13], [15] provide a survey and a comparison of various jobshop scheduling methods. Deterministic algorithms as well as approximation and heuristic approaches (including, simulated annealing, genetic algorithms, and ejection chains) to scheduling manufacturing processes are presented and discussed in [3]. In [11], a case with uncertain processing times is studied and an approach based on fuzzy set theory is proposed.

A decision maker will prefer that sequence which optimizes the chosen measure of effectiveness. Sequencing problems occur in flow-shop production systems and in job-shop production systems[2]. In the former, each production order goes across the same set of machine centers. The jobs may be fixed in number or they may arrive over time. In a job-shop production system, jobs flow across machine centers on many different routes. In this paper the flow-shop problems are presented for the case where the job set is fixed in number. Then the job-shop problem is presented for the dynamic case involving the continuous arrival of jobs over time.

2. Flow-Shop Sequencing For Two Machines

The two-machine flow-shop operation presented in this section may be described as follows. Similar production equipment and skills are grouped into two machine or production centers. There exists a set of jobs, all simultaneously available, which are to be processed.Two constraints exist. First, each machine can process only one job at a time. Second, each job can be in process on only one machine at a time.

Formulating the Sequencing Problem. The objective in determining a production sequence for the set of n jobs is to minimize the maximum flow time.

Let

A(i) = processing time (including set up time if any) on the first machine for the ith job.
B(i) = processing time (including set up time if any) on the second machine for the ith job.
F(i) = time at which the ith job is completed.

Each job consists of a pair of times A(i) and B(i), where A(i) is the time on the first machine and B(i) is the time on the second machine. This ordering is the same for each of the n jobs.

The two-machine flow-shop sequencing problem may now be formulated as follows: Given the 2n time values, A(1), A(2), ……A(n), and B(1), B(2), ……B(n), find the ordering of jobs on each of the two machines so that neither of the constraints is violated and the maximum of F(i) is minimized. Therefore,

$$F_{max} \geq \sum_{i=1}^{n} \mathrm{A(i)} + \mathrm{B(n)}.$$

Similarly, the last job cannot be completed in less time than it takes to process each job on machine B plus the time caused by the delay before machine B can begin, since B(1) cannot overlap A(1). Therefore,

$$F_{max} \geq \sum_{i=1}^{n} \mathrm{A(i)} + \mathrm{B(n)}.$$

It will be noted that the sum of A(i)'s and the sum of B(i)'s are direct consequences of the given processing times and are entirely unaffected by the ordering of the jobs. Thus, to reduce F_{max} one can only influence $B(n)$ and $A(1)$by the choice of sequence. Hence, choose the smallest of the set of $2n$ values of $A(i)'s$ and $B(i)'s$. If this value happens to be an $A(i)$, put this job first in sequences so as to make A(1) as small as possible. If the smallest processing time happened to be B(i), this job would go last in sequence so as to make $B(n)$as small as possible. Once the position of one job is determined, the same procedure can be used for the set of (n – 1) remaining jobs.

2.1. Section one [heading of Section]

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus mollis lorem a magna vestibulum condimentum. Phasellus eget sollicitudin nisi. Fusce blandit congue imperdiet. Interdum et malesuada fames ac ante ipsum primis in faucibus. Fusce quis velit ut turpis blandit venenatis. Vivamus feugiat condimentum tortor vel pharetra. Suspendisse id sapien eros. Pellentesque volutpat felis pellentesque orci sollicitudin commodo. Nullam rhoncus, magna feugiat blandit porta, ligula ipsum rutrum sem, posuere pellentesque urna felis in est. Pellentesque euismod, leo quis bibendum mattis, purus arcu dictum felis, quis elementum nibh ipsum eget augue. Nunc commodo porttitor erat, vitae interdum leo sollicitudin nec. Nam consectetur ex id semper aliquam. Suspendisse et varius erat. Nunc interdum tempus orci vitae imperdiet. Mauris a quam a lectus viverra malesuada vitae ut dui. Fusce iaculis erat in tristique condimentum.

2.2. Section two

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus mollis lorem a magna vestibulum condimentum. Phasellus eget sollicitudin nisi. Fusce blandit congue imperdiet. Interdum et malesuada fames ac ante ipsum primis in faucibus. Fusce quis velit ut turpis blandit venenatis. Vivamus feugiat condimentum tortor vel pharetra. Suspendisse id sapien eros. Pellentesque volutpat felis pellentesque orci sollicitudin commodo. Nullam rhoncus, magna feugiat blandit porta, ligula ipsum rutrum sem, posuere pellentesque urna felis in est. Pellentesque euismod, leo quis bibendum mattis, purus arcu dictum felis, quis elementum nibh ipsum eget augue. Nunc commodo porttitor erat, vitae interdum leo sollicitudin nec. Nam consectetur ex id semper aliquam. Suspendisse et varius erat. Nunc interdum tempus orci vitae imperdiet. Mauris a quam a lectus viverra malesuada vitae ut dui. Fusce iaculis erat in tristique condimentum.

3. Results

Flow-shop sequencing problems involving three or more machines require very complex solution procedures. In this section a special three-machine problem is presented which may be converted into an equivalent two-machine problem for solution by the previous procedure. Next, a general three-machine problem is solved using a branch-and-bound technique. Finally, heuristic methods of solution are discussed.

A special three-machine problem. Assume that there are three machines, A,B, and C, in a flow shop. Each job is processed in the order A,B, C and the processing time is smallest on machine B for every job – that is, $A(i) \leq B(i)$ and $C(i) \geq B(i)$. Under these conditions the three-machine can be converted into an equivalent two machine problem with processing times.

$G(i)=A(i)+ B(i)$and $H(i)=B(i)+ C(i)$

General Three-Machine Sequencing Problem. The branch and bound procedure can be used to sequence n jobs through 3 machines[12]. For n jobs there are n! possible sequences. The objective is to find the one with the minimum makespan.For purposes of branching or partitioning the sets of solutions, a partial sequence (one with less than n jobs sequenced) is used to identify the subset of all possible sequences arising out of the partial sequence. A lower bound on each subset is determined such that the makespan for any sequence in the subset is greater than or equal to this bound. After every

branching the subset with the lowest lower bound is chosen for further exploration. The procedure terminates when a complete sequence is obtained for which the makespan is equal to or less than the lower bounds on all unexplored subsets. The completed sequence is then optimal.

In the three-machine makespan problem, each node represents a sequence of between 1 and n jobs. Consider a node corresponding to partial sequence y, where y contains a particular subset of the n jobs of sizer. Let TA(y), TB(y) and TC(y) be the times at which machines A, B, and C, respectively, complete processing the last of the jobs in the subset

$$LB(y) = \max\left\{\begin{array}{l} \mathrm{TA(y)} + \sum_{\breve{y}} \mathrm{A(i)} + \min_{\breve{y}}[\mathrm{B(i)} + \mathrm{C(i)}] \\ \mathrm{TB(y)} + \sum_{\breve{y}} \mathrm{A(i)} + \min_{\breve{y}}[\mathrm{C(i)}] \\ \mathrm{TC(y)} + \sum_{\breve{y}} \mathrm{C(i)} \end{array}\right\}$$

Where A(i), B(i), and C(i) are the processing times of the ith job on machines A, B, and C, respectively, and $\breve{y}$is the set of n – r jobs that have not been assigned a position in partial sequence y. LB(y) is a lower bound on the makespan for any node that branches from node y, since all such nodes represent sequences of from r + 1 to n jobs that begin with sequence y.

The branches-and-bound technique is applied as follows:

1. Develop a list of nodes ranked by lower bounds such that the node with the smallest lower bound is first.
2. Retain a set of attributes (partial sequence y, TA, TB, TC, etc.) for each node.
3. Start by listing only the node that has scheduled none of the jobs.
4. Update the list recursively as described below until an optimum is obtained.
5. Remove the first node from the list.
6. Create a new node for every job that the "just=removed" node has not yet scheduled. This is done by attaching the unscheduled job to the end of the sequence of scheduled jobs.
7. Compute the lower bounds and other attributes for those newly created nodes and insert them ranked on the list.
8. Go back to step (5). The first time that a node which has scheduled all n jobs is first on the list the problem is solved. The node's sequence is an optimal one.

Flow-shop Sequencing on a Machines. A common characteristic of all algorithms for the solution of sequencing problems is the magnitude of the computations required. Although most heuristic approaches involve much less computation than complete enumeration, the required time increases rapidly as the size of the problem increases. Consequently, practical size-sequencing problems are beyond the reach of total enumeration and must be subjected to heuristic methods of solution. However, heuristic methods do not guarantee an optimal solution.

Consider the heuristic solution procedure below.

Let

$$T_{ij}\ \begin{cases} i = 1, \ldots\ldots\ldots, n \\ j = 1, \ldots\ldots\ldots, n \end{cases}$$

represent the processing time for the *i*th job on the *j*th machine in an *n*-job, m-machine, makespan sequencing problem. Then p auxiliary *n*-job, two-machine problems can be defined. In the *k*th auxiliary problem, where k is the sequence number equal to or less than *p*,

$$\theta_{i1^K} = \sum_{j=1}^{k} T_{ij} = processing\ time\ for\ the\ ith\ job\ on\ he\ first\ machine.$$

$$\theta_{i2^K} = \sum_{j=m+1-k}^{k} T_{ij} = processing\ time\ for\ the\ ith\ job\ on\ he\ second\ machine.$$

The result is a set of sequences $S_1, S_2, \ldots\ldots\ldots, S_p$ for the p auxiliary problems. The best sequence of the set would be chosen.

4. Shop Sequences Problem

In this section a relatively more complex production system known as the job shop is presented. The job-shop system may be described as follows. Similar production facilities and skills are grouped and constitute a finite number of machine centers. Each committed production order or job is either awaiting release to the first machine center on its route or it is in process. The route for each job involves n downstream machine centers and is specified by a routing sheet. The routes for all jobs are determined by technological processing considerations and constitute a network of interwoven paths. Completion of a given job is a composite task requiring the completion of a finite number of subtasks, each consuming scarce machine time.

An important part of the production management task is the sequencing of jobs at machine centers to achieve some desired performance objective. This section assumes that management is primarily concerned with the due date performance of completed production orders. Because of this, time is the primary variable of interest. Thus, we will use a daily production calendar for time which consists of the positive set of integers increasing with time and without bound.

An Urgency Factor Model. Order flow time through the *j*th machine center is uncertain and may be viewed as a random variable. A gross classification of its element would be move time, queue time, setup time, and processing time.

If t_j is used to designate flow time across the *j*th machine center, then total flow time for the *i*th order may be expressed as $T_i = \sum t_j$ for $j = 1,2,\ldots\ldots\ldots\ldots, n$. Since t_j is a random variable, it follows that Ti will also be a random variable.

Assuming that the flow time across machine centers are independently distributed random variables with means μ_j and variances σ^2_j, the distribution of T_i will have parameters.

$$\mu_j = \sum_j \mu_j$$

$$\sigma_{i^2} = \sum_j \sigma_{i^2} \ for\ j = 1,2, \ldots\ldots\ldots\ldots, n.$$

Further, it can be assumed that T_i is distributed approximately normal due to a version of the central limit theorem.

Regardless of the position of the order in the production system, there is a certain upstream history and a certain downstream future that has a bearing on the probability of completing the order by its due date. If the current date is designated C, the time in days remaining before the due date for the *i*th order

is $Di - C$. For an order at machine center $k(k = 1,2,\ldots\ldots,n)$ the expected flow time before completion will be $\sum n_{j-k}\mu_j$. The flow-time variance will be $\sum n_{j-k}\sigma^2_j$. The factor

The relative urgency of orders in each queue is based solely upon the time remaining before due date and the statistical properties of downstream flow time. Although the distribution of flow time will deviate from normality as $k\ldots\ldots\ldots n$,this is important only to the extent that the flow-time distributions differ for orders being sequenced. Of course, if t_j is normal, total flow time will be normal regardless of the number of remaining downstream machine centers.

5. Comparison Of Job-Shop Sequencing Rules

Rules in the first class are completely independent of any characteristics of the job. Sequencing rules that depend on both job and shop characteristics fall in the third class.

The simplest example of a rule that does not depend on any characteristic of the job or the shop is the first-come, first-served (FCFS) rule. This sequencing rule assigns priority to jobs in the order in which they join the queue at a machine center. Symbolically, if P_{ij} is the priority of the ith job behind the jth machine center and s_j is the set of integers denoting the jobs in queue behind machine center j, in arrival order, then for this rule

$$\max\{P_{ij}\} = \min\{s_j 1\}$$

Where $s^1{}_j$is the first elementof the set s_j. Obviously, the assignment of priority by this rule in no way depends on either job or shop characteristics.

A very useful rule known as the **Shortest Processing Time** (SPT) rule is an example of a sequencing policy which depends on a job characteristic but does not depend on any characteristic of the shop. This sequencing rule assigns the highest priority to that job in the queue having the shortest processing time at the machine center in question. If $t_{ij}$is the processing time of the ith job at the jth machine center, then for this rule

$$\max\{P_{ij}\} = \min\{t_{ij}\}.$$

The assignment of priority by this rule depends on the job characteristic of processing time but does not depend on any shop characteristic.

Probability sequencing (PS) as developed in the proceeding section is an example of a sequencing rule that depends on both job and shop characteristics. Priority s assigned to jobs at each machine center in accordance with

$$\max\{P_{ij}\} = \min\{z_{ij}\},$$

Where z_{ij} is the urgency factor for the ith job at the jth machine center. This sequencing rule assigns priorities that depend on both job and shop characteristics as was indicated in the previous section.

Job-Shop Simulation. Comparison of job-shop sequencing rule can be accomplished only by use of digital simulation. Such simulation permits indirect experimentation in a situation where direct experimentation is virtually impossible. The simulator models of a production system with machine centers from which routes for jobs are selected. A job entering the system joins the queue behind the fist machine center on its routing. When the machine is free, a job is selected in accordance with a sequencing rule and is allowed to occupy the machine for a time equal to its predetermined processing time. When the job is completed, it moves into the queue behind the next machine center on its routing. A machine center is idle if no job is in process at that center. A job is complete when it has been processed through all centers on its route.

References

[1] A. El-Bouri, N. Azizi and S. Zolfaghari, "A Comparative Study of aNew Heuristic Based on Adaptive Memory Programming and SimulatedAnnealing: The Case of Job Shop Scheduling," *European Journal ofOperational Research*, 2007, 17 pp., in press.

[2] A. Jain and S. Meeran, "Deterministic Job-Shop Scheduling: Past,Present and Future," *European Journal of Operational Research*, vol.113, pp. 390-434, 1999.

[3] C. H. Papadimitriou and J. N. Tsitsiklis. On stochastic scheduling with in-tree precedenceconstraints. *SIAM Journal on Computing*, 16:1–6, 1987.