OMEGA The Int. Jl of Mgmt Sci. Vol. 11, No. 1, pp. 91-95, 1983
Printed in Great Britain.

0305-0483/83/010091-05$03.00/0

A Heuristic Algorithm for the m-Machine, n-Job Flow-shop Sequencing Problem

MUHAMMAD NAWAZ

University College of Engineering, Pakistan

E EMORY ENSCORE Jr

INYONG HAM

The Pennsylvania State University, USA

(*Received April 1982; in revised form August 1982*)

In a general flow-shop situation, where all the jobs must pass through all the machines in the same order, certain heuristic algorithms propose that the jobs with higher total process time should be given higher priority than the jobs with less total process time. Based on this premise, a simple algorithm is presented in this paper, which produces very good sequences in comparison with existing heuristics. The results of the proposed algorithm have been compared with the results from 15 other algorithms in an independent study by Park [13], who shows that the proposed algorithm performs especially well on large flow-shop problems in both the static and dynamic sequencing environments.

INTRODUCTION

THE SEQUENCING problem is the problem of defining order (rank, priority etc.) over a set of jobs (tasks, items, etc.) as they proceed from one machine (processor) to another. Thus, the sequencing problem involves the determination of the relative position of job i to all other jobs. These types of problems occur in many different environments. They exist whenever there is a choice as to the order (rank) in which a number of tasks (jobs) can be performed. A problem could involve: jobs in a manufacturing plant, aircraft waiting for landing clearances or programs to be executed at a computing center. Since Johnson's [9] solution of the two-machine and special three-machine flow-shop problems, several algorithms have been developed for sequencing n jobs on m machines to minimize makespan in the flow-shop. In all these algorithms, several restrictive assumptions are made, such as the simultaneous availability of all the jobs and the machines, deterministic process times, etc. These assumptions, though far from the real situation, greatly simplify the problem. A complete list of these assumptions is given in Dudek & Teuton [6].

The search for a solution to the problem of finding an optimal or near optimal sequence has yielded both exact and approximate solution techniques. The exact techniques [2, 4, 8, 10] solve the problem in principle, but, in most of the cases, the computation time and the memory required to keep track of the calculations is prohibitive even for small problems. On the other hand, heuristic algorithms, though they do not necessarily provide the optimal solution to the problem, are for the most part an efficient and economical way of getting a good solution to the problem [13].

Palmer [12], in 1965, proposed a slope order

index to sequence n jobs on m machines based on the process times of jobs on different machines. Then in 1971, Gupta [7] suggested another algorithm which is similar to Palmer's except that he defined his slope index in a slightly different manner. Campbell, Dudek & Smith (CDS), in 1970, proposed a procedure [3] which is a heuristic generalization of Johnson's algorithm. This procedure generates a set of $m - 1$ artificial two-machine problems from the original m-machine problem, each of which is then solved by Johnson's algorithm. Many other algorithms have been suggested by researchers [2, 5, 14] to find a near optimal solution for the flow-shop problem. In a recent study by Setiaputra [15], five well known heuristics [3, 5, 7, 12, 14] were statistically evaluated for performance. Based on the Keul–Newman multiple-ranking procedure, the CDS heuristic is shown to be superior.

In the following section, a new curtailed-enumeration algorithm is proposed for the flow-shop problem.

THE PROPOSED ALGORITHM

The flow-shop sequencing problem can be stated as follows:

Given n jobs to be processed on m machines in the same order, the process time of job i on machine j being $t_{i,j}$ $(i = 1, 2, \ldots, n;\ j = 1, 2, \ldots, m)$, find the sequence of jobs such that the total elapsed time (makespan) is minimized.

The flow-shop sequencing problem as presented above is a combinational search problem with $n!$ possible sequences. If one could enumerate all $n!$ sequences, the sequences with minimum total completion time could be identified, but this procedure is quite expensive and impractical for large n.

The proposed algorithm is based on the assumption that a job with more total process time on all the machines should be given higher priority than a job with less total process time. An overview of the proposed algorithm can be stated as follows.

The two jobs with the highest total process times are selected from the n-jobs. The best partial sequence for these two jobs is found by an 'exhaustive search', i.e. considering the two possible partial schedules. The relative positions of these two jobs with respect to each other are fixed in the remaining steps of the algorithm. Next, the job with the third highest total process time is selected and the three partial sequences are tested in which this job is placed at the beginning, middle and end of the partial sequence found in the first step. The best partial sequence will fix the relative positions of these three jobs for the remaining steps. This process is repeated until all jobs are fixed and a complete sequence is found. The number of enumerations in the algorithm is:

$$\frac{n(n+1)}{2} - 1$$

of which n enumerations are complete sequences and the rest are partial sequences.

The following is the step by step procedure for the curtailed-enumeration algorithm:

Step 1. For each job i calculate

$$T_i = \sum_{j=1}^{m} t_{i,j}$$

where $t_{i,j}$ is the process time of job i on machine j.

Step 2. Arrange the jobs in descending order of T_i.

Step 3. Pick the two jobs from the first and second position of the list of Step 2, and find the best sequence for these two jobs by calculating makespan for the two possible sequences. Do not change the relative positions of these two jobs with respect to each other in the remaining steps of the algorithm. Set $i = 3$.

Step 4. Pick the job in the ith position of the list generated in Step 2 and find the best sequence by placing it at all possible i positions in the partial sequence found in the previous step, without changing the relative positions to each other of the already assigned jobs. The number of enumerations at this step equals i.

Step 5. If $n = i$, STOP, otherwise set $i = i + 1$ and go to Step 4.

NUMERICAL ILLUSTRATION

To illustrate the procedure outlined above,

TABLE 1. OPERATION TIME MATRIX

		Machines (m)				
		1	2	3	4	5
Jobs (m)	1	5	9	8	10	1
	2	9	3	10	1	8
	3	9	4	5	8	6
	4	4	8	8	7	2

the four job, five machine flow-shop problem given in Table 1 is solved.

Step 1. $T_1 = 5 + 9 + 8 + 10 + 1 = 33$,
$T_2 = 9 + 3 + 10 + 1 + 8 = 31$,
$T_3 = 9 + 4 + 5 + 8 + 6 = 32$,
$T_4 = 4 + 8 + 8 + 7 + 2 = 29$.

Step 2. 1, 3, 2, 4.

Step 3. Pick job 1 and job 3, and find the optimal partial sequence for these two jobs. This is done in Tables 2 and 3, where it is clear that sequence 3–1 is the best with a makespan = 42. In the next steps, the relative position of job 1 and job 3 should always be 3–1, i.e. job 1 after job 3. Set $i = 3$.

TABLE 2. MAKESPAN FOR PARTIAL SEQUENCE 1–3

		Machines				
		1	2	3	4	5
Jobs	1	5/5	9/14	8/22	10/32	1/33
	3	9/14	4/18	5/27	8/40	6/46

TABLE 3. MAKESPAN FOR PARTIAL SEQUENCE 3–1

		Machines				
		1	2	3	4	5
Jobs	3	9/9	4/13	5/18	8/26	6/32
	1	5/14	9/23	8/31	10/41	1/42

Step 4. Take the job in the third position of the list of Step 2 (job 2) and find the optimal sequence by placing job 2 at all three possible positions in the partial sequence 3–1 obtained in the last step. The makespan of these partial sequences are given in Tables 4, 5 and 6. Three possible combinations tested above show that sequence 3–1–2 (Table 4) is the best with makespan = 50.

TABLE 4. MAKESPAN FOR PARTIAL SEQUENCE 3–1–2

		Machines				
		1	2	3	4	5
Jobs	3	9/9	4/13	5/18	8/26	6/32
	1	5/14	9/23	8/31	10/41	1/42
	2	9/23	3/26	10/41	1/42	8/50

TABLE 5. MAKESPAN FOR PARTIAL SEQUENCE 3–2–1

		Machines				
		1	2	3	4	5
Jobs	3	9 9	4 13	5 18	8 26	6 32
	2	9 18	3 31	10 31	1 32	8 40
	1	5 23	9 32	8 40	10 50	1 51

TABLE 6. MAKESPAN FOR PARTIAL SEQUENCE 2–3–1

		Machines				
		1	2	3	4	5
Jobs	2	9/9	3/12	10 22	1 23	8 31
	3	9/18	4 22	5 27	8 35	6 41
	1	5/23	9/32	8 40	10 50	1 51

Step 5. $i \neq n$, hence $i = 3 + 1 = 4$, and go to step 4.

Step 4. Pick the job in the fourth position of the list of Step 2 (job 4) and find the optimal sequence by placing job 4 in the four possible positions of the partial sequence 3–1–2 found in the last Step 4. Calculations of the makespans are given in Tables 7 to 10. The

TABLE 7. MAKESPAN FOR SEQUENCE 3–1–2–4

		Machines				
		1	2	3	4	5
Jobs	3	9/9	4/13	5 18	8 26	6/32
	1	5/14	9/23	8 31	10 41	1/42
	2	9/23	3/26	10 41	1 42	8/50
	4	4/27	8/35	8 49	7 56	2/58

TABLE 8. MAKESPAN FOR SEQUENCE 3–1–4–2

		Machines				
		1	2	3	4	5
Jobs	3	9/9	4/13	5 18	8 26	6/32
	1	5/14	9/23	8 31	10 41	1/42
	4	4/18	8/31	8 39	7 48	2/50
	2	9/27	3/34	10 49	1 50	8/58

TABLE 9. MAKESPAN FOR SEQUENCE 3–4–1–2

		Machines				
		1	2	3	4	5
Jobs	3	9/9	4/13	5 18	8 26	6/32
	4	4/13	8/21	8 29	7 36	2/38
	1	5/18	9/30	8 38	10 48	1 49
	2	9/27	3/33	10 48	1 49	8/57

TABLE 10. MAKESPAN FOR SEQUENCE 4–3–1–2

		Machines				
		1	2	3	4	5
Jobs	4	4 4	8 12	8 20	7 27	2 29
	3	9 13	4 17	5 25	8 35	6 41
	1	5 18	9 27	8 35	10 45	1 46
	2	9 27	3 30	10 45	1 46	8 54

sequence 4–3–1–2 yields the minimum makespan of 54 (Table 10).

Step 5. $i = n$, hence STOP.

This problem was solved for the optimal makespan by evaluating all $n! = 24$ sequences and the sequence 4–3–1–2 is optimal. The proposed algorithm made nine enumerations of which four were complete sequences and five were partial sequences. It should be noted that the number of enumerations will increase if ties exist in partial sequences and each tied sequence is examined.

For large flow-shop problems the proposed algorithm is best executed via a computer program. The fortran listing for such a program is given in [11].

EFFECTIVENESS OF THE PROPOSED ALGORITHM

The attempt by the authors to demonstrate the effectiveness of the proposed algorithm was to solve a large number of flow-shop problems with the proposed algorithm and compare the results with those obtained by applying the CDS algorithm. The reason for comparing only with the CDS algorithm was based on results by Setiaputra [15] as previously discussed. A total of 2764 flow-shop problems were solved with the number of machines and jobs being set at various levels between 4 and 25. The process times for the jobs were randomly generated from a uniform distribution with range 1 to 99. The results are given in Table 11. As can be seen, the proposed algorithm performs extremely well.

Since this initial attempt at evaluation, Park [13] has completed a most comprehensive study of flow-shop algorithms. The study evaluated 16 algorithms (the proposed algorithm included). Sensitivity to problem size and computational efficiency were studied. The following statements are taken from Park's results:

(1) "It is clear that NEH (Nawaz-Enscore-Ham) is the least biased and the best-operated of the heuristics tested on the small static flow-shop problems (3 to 9 jobs combined with 4 to 20 machines) with respect to makespan and that CDS is the next best..."

(2) "As in the case of the small size problems, it is clear from the test results that for minimizing makespan in large static problems (15 to 30 jobs combine with 4 to 20 machines), NEH is the least biased and most effective..."

TABLE 11. COMPARISON OF PROPOSED ALGORITHM WITH CDS ALGORITHM

Number of jobs	Number of machines	Number of problems	Number of times solution better by proposed algorithm	Number of times solution equal by proposed and CDS algorithms	Number of times solution better by CDS algorithm
4	4	200	41	153	6
4	5	200	43	152	5
4	6	200	53	142	5
4	20	100	14	67	19
5	6	200	77	105	18
5	8	200	66	115	19
5	10	200	72	98	30
5	20	100	38	43	19
6	8	200	103	67	30
6	10	200	95	75	30
6	12	200	95	70	35
6	20	100	55	17	28
10	10	200	185	10	5
10	15	200	175	9	16
10	20	200	176	10	14
15	15	25	25	0	0
20	20	25	23	0	2
25	25	14	14	0	0

(3) "The results of computation times (for large problems) indicates... NEH is less attractive..."

Statements 1 and 2 indicate a clear dominance of the proposed algorithm. With regard to statement 3, it should be noted that the average computation time for the proposed algorithm on the large problems was only 0.3062 secs. This is large when compared to the 0.0081 secs and 0.093 secs of the Gupta algorithm and the Bonney & Gundry algorithm, respectively. However, 0.3062 secs certainly does not represent excessive computer time. The CDS algorithm averaged 0.0885 secs.

CONCLUSIONS

A new curtailed-enumeration algorithm for solving the m-machine, n-job flow-shop problem has been represented in this paper. The algorithm is simple to understand and apply. As with most heuristic algorithms used to solve the flow-shop problem, the use of the computer is recommended when solving problems with large n.

For the range of jobs (3 to 30) and machines (4 to 25), the proposed algorithm has been shown to be superior to the CDS algorithm and other established algorithms. The evaluations were carried out independently by Setiaputra [15] and Park [13]. Even though no attempt was made at evaluating the proposed algorithm with large numbers of machines and jobs (m, $n \geq 100$), it is felt that it would continue to perform well. There is an exception, however. If the number of machines greatly exceeds the number of jobs, then it is expected that the CDS algorithm would outperform the proposed algorithm since the former's effectiveness is dependent on the number of machines and the latter's is dependent on the number of jobs.

The proposed algorithm's solution time is dependent on the number of jobs considered in the flow shop. The minimum number of partial and complete sequences considered by the algorithm is

$$\frac{n(n+1)}{2} - 1.$$

The average computer solution time for n ranging between 15 and 30 is 0.3 secs. Even though this time is not excessive, it is much larger than the 0.09 secs for the CDS algorithm.

Overall, it is felt that the proposed algorithm performs exceptionally well against established algorithms in solving the flow-shop sequencing problem.

REFERENCES

1. ASHOUR S (1970) An experimental investigation and comparative evaluation of flow-shop scheduling techniques. *Ops Res.* **18**, (3).
2. BAKER KR (1974) *Introduction of Sequencing and Scheduling*. John Wiley, New York.
3. CAMPBELL HG, DUDEK RA & SMITH ML (1970) A heuristic algorithm for the n job, m machine sequencing problem. *Mgmt Sci.* **16**, (16).
4. CONWAY RW, MAXWELL WL & MILLER LW (1967) *Theory of Scheduling*. Addison–Wesley, Reading, Massachusetts.
5. DANNENBRING DG (1977) An evaluation of flow-shop sequencing heuristics. *Mgmt Sci.* **23**, (11).
6. DUDEK RA & TEUTON OF (1964) Development of M stage decision rule for scheduling 'n' jobs through 'm' machines. *Ops Res.* **12**, (3).
7. GUPTA JND (1971) A functional heuristic algorithm for the flow-shop scheduling problem. *Op. Res. Q.* **22**, (1).
8. IGNALL E & SCHARGE LE (1965) Application of the branch and bound technique to some flow shop scheduling problems. *Ops Res.* **13**, (3).
9. JOHNSON SM (1954) Optimal two and three-stage production schedules with set-up times included. *Naval Res. Logistic Q.* **1**, (1).
10. LOMNICKI L (1965) A branch and bound algorithm for the exact solution of the three-machine scheduling problem. *Op. Res. Q.* **16**, (1).
11. NAWAZ M (1980) *nxm* flow shop sequencing and scheduling problem. Master's thesis, Pennsylvania State University.
12. PALMER DS (1965) Sequencing jobs through a multistage process in the minimum total time—a quick method of obtaining a near optimum. *Op. Res. Q.* **16**, (1).
13. PARK YB (1981) A simulation study and an analysis for evaluation of performance-effectiveness of flowshop sequencing heuristics: a static and a dynamic flowshop model. Master's thesis, Pennsylvania State University.
14. PETROV VA (1966) *Flow Line Group Production Planning*. Business Publications, London.
15. SETIAPUTRA W (1980) A survey of flow-shop permutation scheduling techniques and an evaluation of heuristic's solution methods. Master's thesis, Pennsylvania State University.

ADDRESS FOR CORRESPONDENCE: *Professor E Emory Enscore Jr, Department of Industrial Management and Systems Engineering, The Pennsylvania State University, 207 Hammond Building, University Park, PA 16802, USA.*