Journal of the Operations Research
Society of Japan
Vol. 29, No. 3, September 1986

FLOWSHOP SCHEDULES WITH SEQUENCE DEPENDENT SETUP TIMES

Jatinder N. D. Gupta
Ball State University

(Received June 24, 1985: Final April 30, 1986)

Abstract The flowshop scheduling problem with sequence dependent setup times is considered and shown to be NP-complete. A traveling salesman problem formulation is proposed for the case where jobs are processed continuously through the shop. These results are used to describe an approximate algorithm for the case where limited or infinite intermediate storage space is available to hold partially completed jobs. The effectiveness of the proposed approximate approach is discussed and some empirical results are reported.

1. Introduction

Consider a static flowshop scheduling problem where n jobs are to be processed on M machines in the same technological order. In formulating this problem, it is usually assumed that setup times are sequence independent and are included in the processing times. However, there are several practical situations, where setup times of a job are separable and depend on the immediately preceding job. As an illustration, consider the scheduling problem in the group technology environment. For each family of parts, a large setup time is required to initiate the processing of family parts after which small setup time is required and is dependent on the sequence of jobs preceding a particular family part (job) being processed [10].

When setup times are separable and sequence independent, and infinite intermediate space is available to hold jobs, Johnson's algorithm [13] can be modified to generate a minimal make-span schedule in $O(n\log n)$ computational steps [20] for the two machine case. Further, if the setup times are separable and sequence dependent on only one of the two machines, a dynamic programming approach can be used to minimize the makespan for problems involving about 15 jobs [2, 4]. Heuristic algorithms for minimizing the makespan for the two

machine case are also available [10, 11].

If the setup times are separable and sequence dependent on all machines, mathematical programming and implicit enumeration approaches that are based on its graph-theoretic representation can be modified to solve the problem [6, 9, 17, 18]. However, the computational requirements for the dynamic programming and the branch and bound algorithms are quite excessive even for problems of moderate size.

This paper discusses the flowshop scheduling problem with sequence dependent setup times and shows that the problem is NP-complete. A traveling salesman problem formulation is suggested for the case where the processing of jobs through the shop is continuous. Based on the traveling salesman formulation of the continuous processing flowshop, an approximation solution algorithm is proposed for the flowshop problem with limited or infinite intermediate storage space in the presence of sequence dependent setup times. The effectiveness of the proposed approximate algorithm is empirically evaluated by solving several flowshop problems and the results of this comparison are discussed.

2. Problem Complexity and Formulation

The following theorem shows that the flowshop problem with sequence dependent set up times and any measure of performance which is a function of the processing and setup times is NP-complete.

Theorem 1. The flowshop scheduling problem with sequence dependent setup times is NP-complete.

Proof: The proof follows by restriction [5]. Let the setup and processing times of all jobs on any combination of $(M-1)$ machines be zero. Then, this is a classical single facility problem with sequence dependent setup times and has been shown to be NP-complete [5], implying that the problem under consideration is also NP-complete. Hence the proof of theorem 1.

The results of theorem 1 above are independent of the number of machines and the intermediate storage space available to hold partially completed jobs. Therefore, unlike the cases where setup times are either sequence independent or can be ignored, the problem remains NP-complete even for the two-machine case. Further, the three cases, Viz: continuous processing, infinite intermediate storage available, and the finite storage space available cases are all NP-complete. In fact, for the two machine case, permutation schedules are not necessarily optimal [10]. To simplify the problem, only permutation

schedules are considered in this paper. This assumption does not affect the global optimality of the continuous processing case, as will be shown shortly.

In addition, results of theorem 1 above are independent of the measure of performance used to evaluate schedules. This implies that flowshop problem with sequence dependent setup times remains NP-complete for any measure of performance involving setup times even for the two machine case, with or without the assumption of permutation schedules.

Let $a_m(i,j)$ be the setup time of job j at machine m if job i immediately precedes it, where $i=0$ if no job precedes job j. Further, let $t(i,m)$ be the processing time of job i on machine m. Number the machines such that jobs are processed on machine 1 first, machine 2 second and the machine M last. With this reindexing of machines, consider a partial schedule Pab formed by concatenating job b to an already known partial schedule Pa. Then the completion time of Pab at machine m, $T(Pab,m)$, is given by the following expression:

$$(1) \qquad T(Pab,m) = max[T(Pab,m-1);\ T(Pa,m) + a_m(a,b)] + t(b,m)$$

where $T(\phi,m) = T(Pab,0)=0$ for all P and m.

Let the measure of performance of partial schedule P be $f(P)$ and $h(P,a)$ represents the addition to the measure of performance of concatenating job a to P. That is:

$$(2) \qquad f(Pa) = f(P) + h(P,a)$$

where $f(\phi)=0$.

The measure of performance in (2) above includes several well known optimization criteria for the flowshop scheduling problem. The one used in this paper is the minimization of weighted sum of completion times on all machines. Let the weight assigned to machine m be w_m. Then, equation (2) becomes:

$$f(Pa) = f(P) + \sum_{m=1}^{M} w_m[T(Pa,m) - T(P,m)]$$

If $w_M=1$ and $w_m=0$ for $m=1, 2, \ldots, (M-1)$, then the above criterion reduces to the minimization of makespan and $h(P,a)$ in equation (2) above is given by:

$$h(P,a) = (T(Pa,M) - T(P,M))$$

The flowshop scheduling problem with sequence dependent setup times can now be defined as one of minimizing $f(Pa)$ where P ranges over all possible permutations of $n-1$ jobs not including job a and job a ranges over all possible values from 1 through n and that the processing and intermediate storage restrictions are satisfied at all machines.

3. Continuous Processing Flowshop

When there is no intermediate storage space available to hold partially completed jobs, the flowshop problem may be formulated in two ways: (1) a job waits at the first machine before starting its processing such that it can be continuously processed by all machines [8, 17], or (2) a partially completed job stays at a particular machine until the next one becomes available [14]. In this paper, the continuous case where jobs are delayed before the start of their first operation is considered. The following theorem restricts the number of feasible schedules to $n!$ for this case.

Theorem 2. For a continuous processing flowshop, only permutation schedules are feasible.

Proof: Consider two consecutive machines m_1 and m_2 where jobs are processed first on machine m_1 and then on machine m_2. Assume that job a is processed before job b on machine m_1 and that job b is processed before job a on machine m_2. Then, since all jobs follow the same technological processing order, it follows that job a will have to wait at least $t(b,m_1) + a_{m_1}(a,b) + t(b,m_2)$ on machine m_2 before starting its processing on machine m_2 after being completed on machine m_1. Since the setup times and processing times are all positive, this waiting time cannot be zero. Therefore, in the schedule considered, job a cannot be processed continuously on all machines. Since m_1, m_2, a, and b are all arbitrary, it follows that a feasible schedule has the same sequence of jobs on all machines. Hence the proof of theorem 2.

Theorem 2 closely follows that in [8]. Since Panwalkar et. al. [15] and Szwarc [19] question the proof for $t(a,m)=0$ for some a and some m, comments are in order here. First, flowshop scheduling problem can be formulated in two ways, one where each job is routed through each machine even though the jobs do not necessarily require processing by a specific machine and the other where jobs may skip the machines [1]. In attempting to formulate the flowshop problem, it is usually assumed that there is no skipping of jobs. This is equivalent to saying that transportation links exist only between consecutive machines. Second, the flowshop is independent of the scale parameter implying that if the parameters A and B are so chosen that $t(a,m) = A + Bp(a,m)$ for all a and m such that $p(a,m)$ is non-negative for all a and m but can be zero, the optimal schedule for the original problem remains the same as for the new problem with $p(a,m)$ as its processing times. Therefore, for the formulation of the flowshop that requires routing of jobs through each of the M machines, the results of theorem 1 are applicable for $t(a,m) \geq 0$ and $a_m(a,b) \geq 0$;

$1 \leq a \leq n$ and $1 \leq m \leq M$. Details of this argument are described in [12].

Since there is no intermediate storage space to hold partially completed jobs, jobs must wait before starting their processing at the first machine. Let the delay in the start of job b after completing job a at machine 1 such that job b is continuously processed on all M machines be denoted by $D(a,b)$. The time interval denoted by $D(a,b)$ assumes that jobs are processed as early as possible. With this definition of $D(a,b)$, the following theorem 3 establishes its value as a function of processing and setup time of jobs a and b alone.

Theorem 3. For the continuous processing flowshop where jobs are processed as soon as possible:

$$(3) \qquad D(a,b) = \max \left\{ \max_{2 \leq m \leq M} \left[\sum_{s=2}^{m} t(a,s) - \sum_{s=1}^{m-1} t(b,s) + a_m(a,b) \right]; \; a_1(a,b) \right\}$$

Proof: By definition of continuous processing on all machines, it follows that:

$$(4) \qquad T(Pa,m) = T(Pa,1) + \sum_{s=2}^{m} t(a,s)$$

$$(5) \qquad T(Pab,m) = T(Pab,1) + \sum_{s=2}^{m} t(b,s)$$

Also, from equation (1), the condition of continuous processing is valid if and only if:

$$T(Pab,m-1) - T(Pa,m) - a_m(a,b) \geq 0 \text{ for all } m.$$

Simplifying the above with equations (4) and (5) above, results in:

$$(6) \qquad D(a,b) \geq \sum_{s=2}^{m} t(a,s) - \sum_{s=1}^{m-1} t(b,s) + a_m(a,b) \quad \text{for all } m.$$

However, $D(a,b)$ is at least as large as $a_1(a,b)$, the setup time of job b on machine 1. Therefore, (3) follows from (6), the above requirement and the fact that jobs are processed as early as possible. Hence the proof of theorem 3.

Results of theorem 3 above show that the initial delay in the start of job b after processing job a is a function the processing times of jobs a and b alone. This implies that the problem can be modeled as a traveling salesman problem. To do so, define:

$$d(a,b) = [D(a,b) + t(b,1)] \sum_{m=1}^{M} w_m \quad ; \text{ for } a=0, 1, \ldots, n \text{ and } b=1, 2, \ldots, n.$$

$$d(a,a) = \infty \tag{7}$$

$$d(b,0) = \sum_{m=2}^{M} w_m [\sum_{s=2}^{m} t(b,s)]$$

With the above distances, the following theorem can be stated:

Theorem 4. The schedule $S=(a_1, a_2, \ldots\ldots, a_n)$ is optimal to the continuous processing flowshop scheduling problem if and only if the tour $(0, a_1, a_2, \ldots\ldots, a_n, 0)$ is optimal to the traveling salesman problem with distance matrix described by Equation (7) above.

Proof: The length, L, of the tour $(0, a_1, a_2, \ldots\ldots, a_n, 0)$ is given by:

$$L = d(0,a_1) + \sum_{i=1}^{n-1} d(a_i, a_{i+1}) + d(a_n, 0)$$

Simplifying the above expression with Equation (7) yields:

$$L = w_1 A + \sum_{m=2}^{M} w_m [A + \sum_{s=2}^{m} t(a_n, s)] \tag{8}$$

where:

$$A = D(0, a_1) + \sum_{i=1}^{n} t(a_i, 1) + \sum_{i=1}^{n-1} D(a_i, a_{i+1}) \tag{9}$$

Now a closer look at (9) reveals that $A=T(S,1)$, the completion time of schedule S at machine 1. Therefore, recursive use of Equations (1) and (4) with Equations (8) and (9) results in:

$$L = \sum_{m=1}^{M} w_m [T(S,m)] \tag{10}$$

Thus, the total length of the tour $(0, a_1, a_2, \ldots\ldots, a_n, 0)$ in Equation (10) equals the weighted sum of completion times on all machines for schedule $S = (a_1, a_2, \ldots\ldots, a_n)$. Therefore, an optimal schedule for the flowshop problem is an optimal tour to the traveling salesman problem (and vice versa) provided the traveling salesman is stationed at city 0 and returns to city 0 after visiting all the n cities. Hence the proof of theorem 4.

While the theorem 1 above showed that the problem is NP-complete, the traveling salesman formulation of the problem does have some advantages. *First*, there are several efficient approximate solution procedures available

to solve the traveling salesman problem (see Parker and Reardon [16] for a recent review), thus providing several approximate solution procedures for the solution of the continuous processing flowshop scheduling problem with sequence dependent setup times. *Second*, worst case analysis of approximate solution procedures for the traveling salesman problem is sometimes available indicating that performance guarantee for solution procedures for the flowshop problem under consideration is possible. *And finally*, probabilistic analysis for the performance of several approximate algorithms for the traveling salesman problem are now available which can be used for the flowshop problem under consideration.

4. Flowshop with Limited or Infinite Intermediate Storage

When there is limited or infinite capacity available to hold partially completed jobs, the results of Theorem 2 no longer hold and non-permutation schedules are feasible. In fact, even for the two machine case, permutation schedules are not necessarily optimal even for the makespan criterion [10]. Therefore, the no-passing assumption (that only permutation schedules are considered) is active even for the two machine case.

The solution of this problem can be approached by implicit enumeration techniques, like the branch and bound procedures [9, 18]. However, these implicit enumeration techniques do not provide a practical approach to solve the problem because of their exponential nature of the computational effort. Realizing this, the traveling salesman approach to the continuous processing flowshop is adopted to generate approximate schedules for this case as well. This is done by finding an approximate solution to the traveling salesman problem using a suitable approximate algorithm. This is especially relevant in view of the fact that the schedules for the two cases are not that far apart from each other without setup times [3]. Therefore, the approximation should work quite well.

The proposed approximate algorithm for the limited or infinite storage space flowshop, therefore, may be described as follows:

1: *Problem Formulation*: Formulate the traveling salesman problem by using the distances given by Equation (7).

2: *Approximate Solution*: Solve the corresponding traveling salesman problem in step 1 using an appropriate approximate procedure like the "closest city" heuristic. Accept the resultant schedule as an approximate solution to the flowshop problem.

The appropriate approximate algorithm for the traveling salesman problem in step 2 above can be selected from several available approximate solution procedures on the basis of the their computational effort and the relative accuracy of the solution obtained through their use. (See Parker and Reardon [16] for a recent review of the traveling salesman problem solution procecedures).

To measure the effectiveness of the proposed approximation, let:

$$e_1 = \frac{\text{opt(continuous)} - \text{opt(infinite)}}{\text{opt(infinite)}}$$

$$e_2 = \frac{\text{Approx(continuous)} - \text{opt(continuous)}}{\text{opt(continuous)}}$$

$$e_3 = \frac{\text{Approx(infinite)} - \text{opt(infinite)}}{\text{opt(infinite)}}$$

Then, since Approx(continuous) is greater than or equal to Approx(infinite), it is easy to see that:

$$e_3 \leq e_1 + e_2 + e_1 e_2$$

Thus, knowing the worst case error bound for the traveling salesman heuristic and the worst case difference between the optimal values of the objective functions for the two problems, worst case error bounds can be found.

5. Computational Experience

Attempts to find error bounds for the worst case of the problem were not successful. Therefore, considerable computational experimentation was conducted to test the validity of the above approximation for the case with infinite storage space. This case was selected for experimentation because it reflects the other extreme of the continuous processing case. For this purpose, the proposed approximate algorithm was programmed in FORTRAN to solve 600 problems ranging from 3 to 7 jobs and 2 to 7 machines with the objective of minimizing makespan. Further, initial setup time at each machine was assumed to be sequence independent. The initial machine setup times and the processing times of these problems were generated from a uniform distribution in the range (1,99). The sequence dependent setup times of all jobs were generated from a uniform distribution in the range (1,9). Each of these

problems was solved by using the proposed approximate algorithm described in Section 4 above. The traveling salesman problem representing the continuous processing flowshop problem in Section 4, represented by Equation (7), was solved to optimality using an existing search algorithm [7]. The percent deviation of the approximate makespan thus found from the optimum makespan, found through complete enumeration, was computed.

Based on 20 problems of each size, the following statistics were collected:

n_1 : number of times approximate makespan was optimal.

n_2 : number of times the deviation of approximate makespan from the optimal makespan was greater than zero but less than or equal to 3 percent.

n_3 : number of times the deviation of approximate makespan from optimal makespan was greater than 3 percent but less than or equal to 5 percent.

MIN: minimum percent deviation of the makespan from optimal makespan.

AVR: average percent deviation of the approximate makespan from optimal makespan.

MAX: maximum percent deviation of the approximate makespan from optimal makespan.

Table 1 depicts these summary statistics for the 30 sets of 20 problems each.

From Table 1, it is clear that the proposed algorithm finds near optimum solutions in many cases, with percent deviation of makespan from optimum makespan often being less than 5 percent. Therefore, it may be concluded that the proposed approximate algorithm is quite effective in minimizing the makespan for the flowshop scheduling problem with infinite storage space in the presence of sequence dependent setup times.

To test the difference between the optimal makespan of the continuous and the infinite storage cases, the percent deviation of the optimal makespan for the continuous case from the infinite storage space case optimal makespan was computed. Table 2 shows these results where n_1, n_2, n_3 etc. need to be interpreted in terms of the percent difference between two optimal makespans.

Table 1 Percentage Deviation from Optimum Makespans

n	M	n_1	n_2	n_3	AVR	MIN	MAX
3	2	15	5	0	0.3552	0.0	2.6906
4	2	9	8	3	0.8963	0.0	3.1142
5	2	3	14	3	1.4590	0.0	4.6703
6	2	3	14	0	1.9972	0.0	6.9307
7	2	2	12	3	2.9240	0.0	13.5081
3	3	17	3	0	0.2244	0.0	2.3196
4	3	9	7	2	1.4757	0.0	6.9252
5	3	6	12	2	0.8683	0.0	4.8458
6	3	1	8	6	3.8746	0.0	12.2857
7	3	3	10	2	2.6675	0.0	8.7097
3	4	15	1	2	1.1729	0.0	7.5243
4	4	8	9	1	1.1904	0.0	7.0352
5	4	10	5	3	2.0935	0.0	13.5699
6	4	2	11	2	3.2048	0.0	10.7325
7	4	0	7	6	4.1305	0.9050	8.0371
3	5	12	4	2	1.2276	0.0	8.9286
4	5	5	9	3	2.1959	0.0	10.2249
5	5	2	8	6	4.0464	0.0	15.3518
6	5	3	9	1	4.2000	0.0	16.1597
7	5	1	6	3	5.2678	0.0	11.2676
3	6	15	4	0	0.6003	0.0	6.9085
4	6	8	9	2	1.0604	0.0	6.0606
5	6	6	9	1	2.3548	0.0	8.2372
6	6	2	9	2	3.7998	0.0	12.1406
7	6	0	4	4	5.9296	0.3989	13.6296
3	7	15	2	3	0.8146	0.0	4.2802
4	7	6	13	0	1.3090	0.0	6.2409
5	7	3	10	4	3.1581	0.0	10.7198
6	7	0	8	3	4.6118	0.1395	12.9506
7	7	0	4	7	5.1556	1.1364	11.4873

A comparison of Tables 1 and 2 shows that the optimal schedule for the continuous case yields solutions of better quality than the differences in the two optimal makespans. In other words, if e_2=0, actual error is less than e_3 where e_1 is given in Table 2.

Table 2 Percent Deviation of Optimal Makespan for Continuous and Infinite Storage Flowshops

n	M	n_1	n_2	n_3	AVR	MIN	MAX
3	2	11	6	1	1.4195	0.0	10.7623
4	2	5	7	5	2.3866	0.0	7.9585
5	2	1	13	3	2.4020	0.0	6.1856
6	2	2	8	2	4.5940	0.0	14.2857
7	2	2	5	2	5.3575	0.0	13.7466
3	3	13	6	0	1.0890	0.0	13.1868
4	3	8	5	2	3.3226	0.0	15.7738
5	3	0	9	4	3.5182	0.4598	8.0169
6	3	0	1	3	8.4981	2.4017	17.7143
7	3	0	1	6	6.7344	0.6483	20.7595
3	4	11	4	2	2.4871	0.0	23.0469
4	4	1	8	6	3.6995	0.0	13.8191
5	4	0	4	3	7.6415	0.4184	19.5313
6	4	0	1	2	9.1703	2.0305	17.0354
7	4	0	0	2	9.3294	3.6199	16.4794
3	5	8	3	3	3.3784	0.0	12.7358
4	5	3	5	5	4.3692	0.0	12.5000
5	5	1	5	3	6.9060	0.0	18.5501
6	5	0	3	2	9.0049	1.8433	18.2509
7	5	0	1	2	9.1840	1.8617	19.0394
3	6	13	3	2	1.3349	0.0	9.4991
4	6	1	12	3	2.8258	0.0	9.0526
5	6	2	4	4	5.1305	0.0	11.6585
6	6	0	1	4	7.7955	1.3575	14.2020
7	6	0	0	1	11.4400	3.5672	21.1852
3	7	10	4	4	1.9615	0.0	11.4458
4	7	2	12	2	2.8393	0.0	8.8388
5	7	0	6	2	6.1105	0.4231	14.0669
6	7	0	2	1	9.0520	1.8939	17.6235
7	7	0	0	1	10.4501	3.5461	19.3215

The computational time for the proposed approximate algorithm was not measured since it depends on the solution method used to solve the traveling salesman problem. The transformation of the flowshop scheduling problem to a traveling salesman problem can be accomplished in polynomially bounded com-

putational effort. Therefore, if the approximate algorithm for the traveling salesman problem is polynomially bounded, so will the proposed approximate algorithm for the flowshop problem.

6. Conclusions

This paper has discussed the flowshop scheduling problem with sequence dependent setup times and shown that the problem is NP-complete for the continuous processing and the limited or infinite or storage space available cases. A traveling salesman formulation of the continuous processing case has been proposed and used to find approximate solutions to the limited or infinite storage space cases as well. Computational results indicated that the proposed approximate algorithm was relatively effective in finding the optimal or near optimal solutions. While the finite storage space case was not considered during the computational phase, it is felt that the proposed approximate algorithm should perform better for this case because of its being somewhere in the middle of the two cases considered. Further, some computational simplifications, like those indicated by Szwarc [19] for the sequence independent case can be used to decrease computational effort of the proposed algorithm.

Acknowledgements

The author appreciates thoughtful comments of two anonymous referees which improved the clarity of this paper.

References

[1] Baker, K. R.: *Introduction to Sequencing and Scheduling*, John Wiley and Sons, Inc., New York, N. Y. (1974).

[2] Bellman, R., A. O. Esogbue and I. Nabeshima: *Mathematical Aspects of Scheduling and Applications*, Pergamon Press, In., New York, N. Y. (1982).

[3] Bonney, M. C., and S. W. Gundry: "Solutions to the Constrained Flowshop Sequencing Problem", *Operational Research Quarterly*, Vol.27, No.4, pp. 869-883 (1976).

[4] Corwin, B. D., and A. O. Esogbue: "Two-Machine Flowshop Scheduling Problems with Sequence Dependent Setup Times: A Dynamic Programming Approach," *Naval Research Logistics Quarterly*, Vol.21, pp.515-524 (1974).

[5] Garey, M. R., and D. S. Johnson: *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman and Company, San Francisco, CA (1979).

[6] Gupta, J. N. D.: "A Review of Flowshop Scheduling Research," in L. P. Ritzman et al (Eds) *Disaggregation: Problems in Manufacturing and Service Organizations*, Martinus Nijhoff, The Hague (1979).

[7] Gupta, J. N. D.: "A Search Algorithm for the Traveling Salesman Problem", *Computers and Operations Research*, Vol.5, No.4, pp.243-248 (1978).

[8] Gupta, J. N. D.: "Optimal Flowshop Scheduling with no Intermediate Storage Space", *Naval Research Logistics Quarterly*, Vol.27, No.2, pp. 235-243 (1976).

[9] Gupta, J. N. D.: "A Search Algorithm for the Generalized Flowshop Scheduling Problem", *Computers and Operations Research*, Vol.2, No.2, pp.83-90 (1975).

[10] Gupta, J. N. D., and W. P. Darrow: "Two-Machine Flowshop Scheduling Problems with Sequence Dependent Setup Times," Paper presented at the ORSA/TIMS Meeting, Orlando Florida, November 7-9 (1983).

[11] Gupta, J. N. D., and W. P. Darrow: "Approximate Schedules for the Two-Machine Flowshop with Sequence Dependent Setup Times", *Indian Journal of Management and Systems*, Vol.1, No.1, pp.6-11 (1985).

[12] Hefetz, N., and I. Adiri: "The Influence of Missing Operations on Scheduling Problems", *Naval Research Logistics Quarterly*, Vol.29, No.3, pp.535-539 (1982).

[13] Johnson, S. M.: "Optimal Two- and Three-stage Production Schedules with Setup Times Included," *Naval Research Logistics Quarterly*, Vol.1, pp. 61-68 (1954).

[14] Levner, E. M.: "Optimal Planning of Parts' Machining on a Number of Machines", *Automation and Remote Control*, No.12, pp.1972-1979 (1969).

[15] Panwalkar, S. S., M. L. Smith and C. R. Woollam: "Counterexamples for Certain Flowshop Problems", *Naval Research Logistics Quarterly*, Vol.28, No.2, pp.339-340 (1981).

[16] Parker, R. G., and R. L. Reardon: "The Traveling Salesman Problem: An Update of Research", *Naval Research Logistics Quarterly*, Vol.30, No.1, pp.69-96 (1983)

[17] Srikar, B. N., and S. Ghosh: "An MILP Model for the N-job-M-Stage Flowshop with Sequence Dependent Setup Times", Working Paper No.WPS 84-32, The ohio State University, Columbus, Ohio (1984).

[18] Srikar, B. N., and S. Ghosh: "An Optimal Algorithm to Minimize Makespan in Sequence Dependent Flowshop", Working Paper No.WPS 84-57, The Ohio

State University, Columbus, Ohio (1984).

[19] Szwarc, W.: "A Note on the Flowshop Problem without Interruptions in Job Processing," *Naval Research Logistics Quarterly*, Vol.28, No.4, pp.665-669 (1981).

[20] Yoshida, T., and K. Hitomi: "Optimal Two-Stage Production Scheduling with Setup Times Separated," AIIE *Transactions*, vol.11, pp.261-263 (1979).

Jatinder N. D. GUPTA: Department of
Management Science,
Ball State University, Muncie,
Indiana 47306