Proceedings of the 2000 Winter Simulation Conference
J. A. Joines, R. R. Barton, K. Kang, and P. A. Fishwick, eds.

SCHEDULING FLOW-SHOPS WITH LIMITED BUFFER SPACES

Michael X. Weng

Department of Industrial and Management Systems Engineering
University of South Florida
4202 East Fowler Avenue
Tampa, FL 33620, U.S.A.

ABSTRACT

The work described in this paper attempts to validate the implicit assumption in traditional flow shop scheduling research that there is a buffer of infinite capacity between any two adjacent machines. The modified NEH (Nawaz, Encore and Ham) algorithm is used to generate an initial permutation schedule which is then improved by tabu search. For any given sequence, a limited equal buffer size is considered in computing job completion times. The scheduling objective is to minimize mean job flowtime. Computational results and analysis are presented. Through these simulation experiments, it was found that the improvement by tabu search can be significant and there is no need for more than 4 buffer spaces between any two adjacent machines. Future research directions are also discussed.

1 INTRODUCTION

This paper addresses the flow shop scheduling problem described as follows. A number of jobs are to be processed on a number of machines. Each job must go through all the machines in exactly the same order. Each machine can process at most one job at any point in time, and each job may be processed on at most one machine at any time. All jobs are ready for processing at time zero. There are limited buffer spaces between machines but there is unlimited buffer capacity before the first machine. The objective is to schedule the given jobs on the given machines such that the mean job flowtime is minimized. It is well known that the mean flowtime flowshop scheduling problem is NP-Hard even if there are only two machines with infinite buffer capacity (Garey, Johnson and Sethi 1976).

Most research on flow shop scheduling assumes unlimited buffer capacity (e.g. Applegate and Cook 1991, Balas 1969, Carlier and Pinson 1989, Ho and Chang 1995, Morton and Pentico 1993, Wang, Chu and Proth 1997). Some authors have considered finite buffers between machines, and machine failures (Conway et al. 1988, Hillier and So 1996, Vouros and Papadopoulos 1998) and concluded that the buffer capacity affects shop performance but the performance improvement diminishes rapidly with increased buffer size. Altiok (1985) developed a Markov chain approach to obtain certain performance measures, for production flow lines with finite buffer storage between unreliable machines. Bloat (1997) implemented a dynamic program to schedule jobs to minimize total blocking time for an automated manufacturing system with buffer in a paced line at a fixed rate. Van Deman and Baker (1973) studied minimizing mean flowtime in the flow shop with no intermediate queues. Zavanella, Agliari and Diligenti (1992) analyzed the buffer saturation phenomenon under various priority rules. Modern search methods such as tabu search and simulated annealing to find good heuristic solutions have also been studied (e.g. Ishibuchi, Misaki and Tanaka 1995, Nowicki and Smutnicki 1996).

Most research involving limited buffers used the Markov chain approach and emphasized developing the relationship between buffer capacity and processing time variation, or identifying the optimal buffer size and allocation, in an attempt to maximize shop throughput which is equivalent to minimizing makespan. On the other hand, the traditional scheduling research ignores the buffer consideration all together (by implicitly assuming infinite buffer capacity). Due to the computational complexity of the flowshop scheduling problem, simple heuristics probably have been the only widely used scheduling methods in practice. To the best of our knowledge, however, the impact of limited buffers on the performances of traditional simple heuristics has not been addressed in the literature. This paper attempts to make a step toward addressing this issue.

There is no doubt that finite buffer capacity will affect the performances of simple heuristics. But, how significant will the impact be? Does there exist a reasonable maximum buffer size beyond which performance improvement will completely diminish? How much can tabu search improve the heuristic solutions? This paper attempts to answer these

questions. The rest of the paper is organized as follows. Section 2 formally states the underlying flowshop scheduling problem. Section 3 presents the NEH heuristics used in this study. Tabu search is described in Section 4. The experimental design is given in Section 5. Section 6 presents the simulation results and analysis. Section 7 concludes the paper.

2 PROBLEM STATEMENT

In a typical static flow shop, a set of n jobs is simultaneously available for being processed on a set of m machines. Without loss of generality, assume that all jobs are available for processing at time zero. Each job j, $j \in J = \{1, 2, \ldots, n\}$, passes through the machines 1, 2, …, m in that order and requires an uninterrupted processing time p_{jk} on machine k, $k = 1, 2, \ldots, m$. Each machine may process the n jobs in any order. If all machines process the n jobs in exactly the same order, the schedule is called a permutation schedule. In this paper, we will focus on permutation schedules. The scheduling objective is to minimize the makespan. It is worth noticing that in this case, a job sequence uniquely determines a permutation schedule since unforced machine idleness is undesirable.

It is assumed that a job may be processed by at most one machine and a machine may process at most one job at any point of time. If a job is completed on machine k and the buffer at machine k+1 is full, the job cannot be unloaded and must stay on machine k until there is available space in the buffer at machine k+1. Let $[i]$ denote the index of the i–th job in a schedule. Then, in case of infinite buffer capacity, job $[i]$ can not start on machine k before it is completed on machine k-1, or before job $[i$-1] is completed on machine k. In case of finite buffer capacity, we let b_k denote the buffer size at machine k. Then, the start of job $[i]$ on machine k may also be blocked if there are already b_{k+1} jobs waiting for machine k+1.

Let C_{jk} denote the completion time of job j on machine k. Then, $C_{[i],k}$ is the completion time of the $[i]$th job on machine k. If the buffer capacity is limited, $C_{[i]k}$ can be computed as follows.

$$C_{[i],k} = p_{[i],k} + \max\{C_{[i],k-1}, C_{[i-b_{k+1}-1],k+1}, C_{[i-1],k}\}. \quad (1)$$

If the buffer capacity is infinite, the above formula is reduced to

$$C_{[i],k} = p_{[i],k} + max\{C_{[i],k-1}, C_{[i-1],k}\}, \quad (2)$$

where $i = 2, 3, \ldots, n$, and $k = 2, 3, \ldots, m$.

The mean flwtime $\overline{F}$ is equal to $(1/n)\sum_j C_{[j],m}$. Our goal is to find a permutation schedule that minimizes $\overline{F}$.

3 HEURISTIC IMPLEMENTATION

As mensioned earlier, many simple heuristics have been developed for the flowshop scheduling problem. The NEH algorithm of Nawaz, Encore and Ham (1988) appears to be the best heuristic for flowshops in minimizing makespan (Talliard, 1990) and mean flowtime (Ho, 1993). We use the modified version of NEH algorithm (Woo and Yim 1998) which is described as follows.

3.1 Modified NEH Algorithm

Step 1. Arrange the jobs in descending order of the sum of processing times.

Step 2. Set k = 2. Pick the first two jobs from the rearranged jobs list and schedule them in order to minimize the mean flowtime as if there are only two jobs. Set the better one as the current solution.

Step 3. Increment k by 1. Generate k candidate sequences by inserting the first job in the remaining job list into each slot of the current solution. Among these candidates, select the best one with the least partial mean flowtime. Update the selected partial solution as the new current solution.

Step 4. If k = n, a schedule (the current solution) has been found and stop. Otherwise, go to step 3.

For any given sequence and given buffer size, job completion times are computed by (1).

4 TABU SEARCH

Tabu search has been used widely in combinatorial optimization (Glover and Laguna 1997). The basic idea is to slightly alter a known (current) solution in a certain manner (called *neighborhood structure*) and take the best alteration as the new current solution. Such altered solutions are called *neighbors* of the current solution. An operation that yields a neighbor is called a *move*. To avoid being trapped at a local optima, the best neighbor that is worse than the current solution is allowed to become the new current solution. To avoid cycling, certain moves are marked as tabu. A tabu move may be allowed if some aspiration criterion is satisfied. This procedure continues until some criterion is met.

The initial current solution is obtained by the modified NEH algorithm. The neighborhood structure considered is the general pairwise interchange (i.e., interchanging jobs i and j, for all i and all $j > i$). In each iteration, the entire neighborhood is searched. Tabu tenure is set to be 7. We used the simple aspiration criterion: A tabu move is overridden if it leads to a schedule that is better than the current best.

The tabu search stops after a maximum number of iterations. Our preliminary runs indicated that tabu search

almost does not find better solutions after around 70 iterations. Therefore, we have set the maximum number of iterations to be 100. That is, tabu search will terminate after 100 iterations.

5 EXPERIMENTIAL DESIGN

In this study we considered 5, 10 and 20 machine cases with 20, 50 and 100 jobs. The generation of processing times is similar to that of Thaillard (1993). In particular, the processing time of each job on each machine follows the uniform distribution U[1, 99]. This represents a balanced flow shop and gives an average job processing time of about 50 on each machine.

Since buffers act as cushions for workload/traffic flow variability, which may result from the inconsistency processing capabilities of machines, the failure or maintenance of machines, or unbalanced job arrivals and fluctuations in processing times, buffers have been employed in most manufacturing systems to enhance their productivity and efficiency. The relevant direct and indirect production costs will increase when the buffer capacities grow. And a larger buffer storage raises the throughput of the system at the expense of more work in process (WIP) inventory (So, 1990). However, low ratios of WIP to throughput are necessary to maintain a competitive production (Conway, 1988). Thus there have been necessities and importance to analyze buffer resources to obtain an optimal buffer capacity for each machine under certain constraints. In this paper we consider the equal buffer capacity case. That is, the buffer capacity between any two adjacent machines is the same. 6 buffer sizes (b = 0, 1, 2, 3, 4, 5, 7) are tested. For each combination of n and m, 10 test problems are generated. For each test problem, all 7 buffer sizes are considered. This leads to a total of 630 test problem instances.

6 COMPUTATIONAL RESULTS

The computational results are presented in table 1, where the last column is the improvement of tabu search over the NEH solutions. That is, Imp. = 100×(NEH-tabu)/NEH %.

From Tables 1-3, we can see that tabu search improves the heuristic solutions by the NEH algorithm by 2% to over 18%. As buffer size increases or as the number of machines increases, tabu search improvement decreases.

Table 1: Mean Flowtime for n = 20 Jobs

m	b	NEH	tabu	Imp. (%)
5	0	876.76	779.57	11.09
	1	763.76	714.91	6.40
	2	755.65	707.92	6.32
	3	755.65	707.92	6.32
	4	755.65	707.92	6.32
	5	755.65	707.92	6.32
	6	755.65	707.92	6.32
10	0	1191.1	1115.91	6.31
	1	1092.45	1055.57	3.38
	2	1083.68	1050.23	3.09
	3	1083.68	1050.23	3.09
	4	1083.68	1050.23	3.09
	5	1083.68	1050.23	3.09
	6	1083.68	1050.23	3.09
20	0	1838.37	1734.17	5.67
	1	1758.31	1689.32	3.92
	2	1755.92	1683.46	4.13
	3	1755.92	1683.46	4.13
	4	1755.92	1683.46	4.13
	5	1755.92	1683.46	4.13
	6	1755.92	1683.46	4.13

Table 2: Mean Flowtime for n = 50 Jobs

m	b	NEH	tabu	Imp. (%)
5	0	1919.8	1622.63	15.48
	1	1599.94	1429.5	10.65
	2	1533.33	1419.14	7.45
	3	1528.17	1410.68	7.69
	4	1528.17	1410.68	7.69
	5	1528.17	1410.68	7.69
	6	1528.17	1410.68	7.69
10	0	2336.84	2077.12	11.11
	1	1981.85	1867.94	5.75
	2	1930.17	1841.26	4.61
	3	1926.07	1839.48	4.50
	4	1925.97	1833.98	4.78
	5	1925.97	1833.98	4.78
	6	1925.97	1833.98	4.78
20	0	3020.31	2787.14	7.72
	1	2699.94	2582.32	4.36
	2	2666.95	2562.45	3.92
	3	2666.95	2556.41	4.14
	4	2666.95	2556.41	4.14
	5	2666.95	2556.41	4.14
	6	2666.95	2556.41	4.14

Table 3: Mean Flowtime for n = 100 Jobs

m	*b*	NEH	tabu	Imp. (%)
	0	3634.71	2973.67	18.19
	1	2975.76	2602.85	12.53
	2	2801.69	2534.06	9.55
5	3	2754.01	2525.13	8.31
	4	2745.64	2522.02	8.14
	5	2745.64	2522.02	8.14
	6	2745.64	2522.02	8.14
	0	4283.15	3777.21	11.81
	1	3445.42	3194.45	7.28
	2	3249.89	3079.74	5.24
10	3	3203.17	3054.37	4.65
	4	3192.54	3047.31	4.55
	5	3189.76	3047.31	4.47
	6	3189.76	3047.31	4.47
	0	5017.18	4634.34	7.63
	1	4237.59	4063.88	4.10
	2	4096.73	4005.65	2.22
20	3	4072.88	4003.91	1.69
	4	4070.59	3988.86	2.01
	5	4070.59	3988.86	2.01
	6	4070.59	3988.86	2.01

It can also be seen that buffer capacity does affect the mean flowtime performance, but the effect diminishes very rapidly as buffer size increases. This can be seen from table 4. Note that the percentage improvement by one buffer unit increase is computed by $[\,\overline{F}(b) - \overline{F}(b+1)\,]/\overline{F}(b)$, where $\overline{F}(b)$ is the mean flowtime when the buffer size is b (= 0, 1, 2, 3, 4, 5). The mean flowtime is significantly reduced when the buffer size is increased from zero to one. The buffer effect dramatically decreases after that. As a matter of fact, for all the problem instances tested, there is no need for more than 4 buffer spaces. On the other hand, the buffer effect increases as the number of jobs increases.

Table 4: Mean Flowtime Improvement (%) by One Buffer Unit Increase

m	*n*	$0 \rightarrow 1$	$1 \rightarrow 2$	$2 \rightarrow 3$	$3 \rightarrow 4$	$4 \rightarrow 5$	$5 \rightarrow 6$
	20	8.29	0.98	0	0	0	0
5	50	11.90	0.72	0.60	0	0	0
	100	12.47	2.64	0.35	0.12	0	0
	20	5.41	0.51	0	0	0	0
10	50	10.07	1.43	0.10	0.30	0	0
	100	15.43	3.59	0.82	0.23	0	0
	20	2.59	0.35	0	0	0	0
20	50	7.35	0.77	0.24	0	0	0
	100	12.31	1.43	0.04	0.38	0	0

7 CONCLUSION REMARKS

The work described in this paper attempted to validate the implicit assumption in traditional flow shop scheduling research that there is a buffer of infinite capacity between any two adjacent machines. The modified NEH algorithm is used to generate an initial permutation schedule which is then improved by tabu search. For any given sequence, a limited equal buffer size is considered computing job completion times. The scheduling objective is to minimize mean job flowtime.

It is found that the impact of limited buffers on the performance of traditional heuristics for static flow shop scheduling is significant only for small buffer sizes and diminishes rapidly as buffer size increases. This indicates that buffer capacity can be treated as infinite once the buffer size reaches a certain critical size. Our simulation results show that this critical buffer size for up to 100 jobs and 20 machines is no more than 4. Using larger buffer sizes will not improve shop performance and should be discouraged since additional costs may incur. The buffer impact is more significant when there are more jobs. Tabu search can improve the NEH heuristic solutions by 2% to 18%. As buffer size increases or as the number of machines increases, tabu search improvement decreases.

A future research area is to study the impact of limited buffers on commonly used dispatching rules in flow shop scheduling with other processing time distributions, with other scheduling objectives such as mean tardiness, and with machine breakdowns. Another area is to extend this research to other shop environments such as unbalanced flow shop, dynamic flow shops and job shops.

REFERENCES

Altiok, T. 1985. Production lines with phase-type operation and repair times and finite buffers. *International Journal of Production research*, 23: 489-498.

Applegate, D. and W. Cook. 1991. A computational study of the job shop scheduling problem. *ORSA Journal on Computing*, 3: 149-156.

Balas, E. 1969. Machine sequencing via disjunctive graphs: An implicit enumeration approach. *Operations Research*, 17: 941-957.

Bolat, A. 1997. Sequencing jobs for an automated manufacturing module with buffer. *European Journal of Operational Research*, 96: 622-635.

Carlier, J. and E. Pinson. 1989. An algorithm for solving the job-shop problem. *Management Science*, 35: 164-176.

Conway, R., W. Maxwell, J.O. McClain, and L.J. Thomas. 1988. The role of work-in-process inventory in serial production lines. *Operations Research*, 36: 229-241.

Garey, M.R., D.S. Johnson, and R. Sethi. 1976. The complexity of flow-shop and job-shop scheduling *Mathematics of Operations Research*, 1: 117-129.

Hillier, F.S., and K.C. So. 1996. On the simultaneous optimization of server and work allocations in production line systems with variable processing times. *Operations Research*, 44: 435-443.

Ho, J. C. and Y.L. Chang. 1995. A new heuristic for the n-job, m-machine flow shop problem. *European Journal of Operational Research*, 52: 194-206.

Ishibuchi, H., S. Misaki, and H. Tanaka. 1995. Modified simulated annealing algorithms for the flow shop scheduling problem. *European Journal of Operational Research*, 81: 388-398.

Morton, T.E. and D.W. Pentico. 1993. *Heuristic scheduling systems*. John Wiley & Sons, New York.

Newaz, M., E.E. Encore, and I. Ham. 1988. A heuristic algorithm for the m-machine, n-job flowshop sequencing problem. *OMEGA*, 11: 91-95.

Nowicki, E. and C. Smutnicki. 1996. A fast tabu search algorithm for the flow shop problem. *European Journal of Operational Research*, 91: 160-175.

So, K.C. 1990. The impact of buffering strategies on the performance of production line systems. *International Journal of Production Research*, 26: 2293-2307.

Taillard, E. 1993. Benchmarks for basic scheduling problems. *European Journal of Operational Research*, 64: 278-285.

Van Deman, J.M., and K.R. Baker. 1973. Minimizing mean flowtime in the flow shop with no intermediate queues. *AIIE Transactions,* 6: 28-34.

Vouros, G. A. and H.T. Papadopoulos. 1998. Buffer allocation in unreliable production lines using a knowledge based system. *Computers and Operations Research*, 25: 1055-1067.

Wang, C., C. Chu, and J.M. Proth. 1997. Heuristic approaches for $n/m/F/\sum C_i$ scheduling problems. *European Journal of Operational Research*, 96: 636-644.

Zavanella, L., A. Agliari, and M. Diligenti. 1992. Dispatching rules and flexible environments: The influence on buffer behavior. *International Journal of Production Research*, 30: 749-772.

AUTHOR BIOGRAPHY

MICHAEL X. WENG is an Assistant Professor in the Department of Industrial and Management Systems Engineering at The University of South Florida since 1995. He got his Ph.D. degree in Industrial Engineering from The Pennsylvania State University in 1994. Prior to joining USF, he worked for Ford New Holland for about a year. His research interest and expertise are in the areas of production planning and scheduling, multi-agent based intelligent scheduling, product and material handling systems, operational modeling and simulation, and applied operations research. He is a member of IIE and INFORMS. His email address is <weng@eng.usf.edu>.