# SCHEDULING A PERMUTATION FLOWSHOP WITH SEQUENCE-DEPENDENT FAMILY SETUPS TO MINIMIZE TOTAL TARDINESS

**Jeffrey E. Schaller, Eastern Connecticut State University, Department Of Business Administration, 83 Windham ST., Willimantic, CT 06250, 860 465 – 5226, SCHALLERJ@EASTERNCT.EDU**
**Jatinder N. D. Gupta, University of Alabama in Huntsville, College of Administrative Science**
**Huntsville, AL 35899, GUPTAJ@UAH.EDU**

## ABSTRACT

This paper presents procedures for scheduling a permutation flowshop with sequence-dependent family setups when the objective is to minimize total tardiness. These procedures are tested on several problem sets. The results show that neighborhood searches that include neighborhoods based on batches of jobs and genetic algorithms are most effective.
**Keywords: Scheduling, Sequencing, Heuristics.**

## INTRODUCTION

In many operations obtaining economies of scale is a critical element in achieving success. Economies of scale are efficiencies in production whereby per-unit production costs increase at a slower rate than production volume. In scheduling efficiencies that lead to economies of scale are gained by grouping similar jobs together. In some settings, the grouping of jobs is a desirable or necessary tactic because of some technological feature of the processing capability. The motivation for grouping sometimes relates to change-over times, or setup times, on the machines. For example, jobs may belong to families where the jobs in each family tend to be similar in some way, such as their required tooling. As a result of this similarity, a job does not need a setup when following another job from the same family, but a known "family setup time" is required when a job follows a member of some other family. This is called a family scheduling model. Typically, there are a large number of jobs, but a relatively small number of families. In some cases the time to setup a machine for a family depends not only on the family to be setup but also on the previous family that was being processed by the machine. This is the case that is considered in this paper.

An important consideration when sequencing and scheduling a set of jobs is completing each job on or before the customer's due date. To address this consideration, this paper seeks to identify and compare methods for sequencing a set of jobs in a permutation flowshop with significant sequence-dependent family setup times that will minimize the total tardiness of the jobs. The tardiness of a job is defined as the completion time of the job minus the due date for the job if the job is completed after the due date and the tardiness is equal to zero if the job is completed before the due date. One approach to sequencing jobs in this environment is to sequence jobs in the same family as a batch (jobs in the same family are sequenced next to each other) to reduce

setup time. The batching of jobs could cause some jobs to be processed before they are needed while at the same time delaying other jobs and causing them to be tardy. A second approach is to stop processing jobs in one family so a job in another family can be processed and completed by its due date. This causes an additional setup to be required and this additional setup increases the overall time to complete jobs that could have the effect of causing jobs that are at the end of the sequence to be very tardy.

Most research on flowshops has assumed the sequence of jobs to process will be the same on each machine. These schedules are referred to as permutation schedules. This is done for two reasons. First it simplifies the computational effort and second it is often not practical to change the sequence of jobs from one machine to the next. In this research only permutation schedules are considered.

Formally, suppose there is a set of n jobs belonging to F setup families to be processed in a flowshop. Let $f_j$, and $d_j$ represent the setup family and the due date of job j (j = 1, …, n) respectively. Let $p_{jm}$, and $C_{jm}$ represent the processing time and completion time of job j (j = 1, …, n) on machine m (m = 1, …, M). Let $S_{jim}$ represent the setup time for job j (j = 1, …, n) on machine m (m = 1, …, M) if job j is processed immediately after job i (i = 1, …, n) and $S_{j0m}$ represent the setup time for job j (j = 1, …, n) on machine m (m = 1, …, M) if job j is processed first in a sequence. The tardiness of job j, $T_j$ is defined as:

$$T_j = \max \{C_{jm} - d_j, 0\}, \text{ for } j = 1,\ldots, n. \quad (1)$$

The objective function, Z, can be expressed as:

$$Z = \sum_{j=1}^{n} T_j. \quad (2)$$

Also, note that if the job to be sequenced in position j is denoted as [j] then

$$C_{[j]1} = C_{[j-1]1} + p_{[j]1} \text{ if } f_{[j]} = f_{[j-1]} \quad (3)$$

$$\text{and } C_{[j]1} = C_{[j-1]1} + S_{[j][j-1]1} + p_{[j]1} \text{ if } f_{[j]} \neq f_{[j-1]}, \quad (4)$$

$$\text{and } C_{[j]m} = \max \{ C_{[j]m-1}, C_{[j-1]m}\} + p_{[j]m} \text{ if } f_{[j]} = f_{[j-1]} \quad (5)$$

$$\text{and } \max \{ C_{[j]m-1}, C_{[j-1]m} + S_{[j][j-1]m}\} + p_{[j]m} \text{ if } f_{[j]} \neq f_{[j-1]} \text{ for } m = 2, \ldots, M. \quad (6)$$

Streams of literature that are most relevant to the problem addressed in this paper are heuristics for minimizing total tardiness in a flowshop and scheduling flowshops with sequence-dependent family setup times. (Vallada and Minella, 2008) reviewed and tested over 40 heuristic methods for the m-machine flowshop problem. Based on their tests (Vallada and Minella, 2008) found that the neighborhood searches developed by (Kim et al., 1996) were the best heuristics and simulated annealing algorithms developed by (Parthasarathy and Rajendran, 1997) and (Hasija and Rajendran, 2004) were the best meta-heuristics. (Vallada and Ruiz, 2010) proposed three genetic algorithms for minimizing total tardiness in permutation flowshops and performed computational tests for these algorithms and other metaheuristics. They found that two of the

genetic algorithms performed the best. (Framinan and Leisten, 2008) developed variable greedy algorithms and found them to be more effective than the procedure developed by (Hasija and Rajendran, 2004) in computational tests. (Tavakkoli-Moghaddam et al, 2007) developed a hybrid multi-objective immune algorithm for a no-wait flowshop with multiple objectives including minimizing weighted tardiness.

(Cheng et al., 2000) provide a review of flowshop scheduling research with setup times. Most of the papers identified in this review consider the objective of minimizing makespan. (Allahverdi et al., 2008) review scheduling problems with setup times. This review includes flowshops with sequence-dependent and sequence-dependent family setup times. (Zhu and Wilhelm, 2006) review scheduling problems with sequence-dependent setup times including flowshops. Recent papers that address flowshops with sequence-dependent setup times with tardiness as part of the objective include: (Ruiz and Stutzle, 2008), (Naderi et al., 2009) and (Ying et al., 2010).

## HEURISTC PROCEDURES

Ten heuristic procedures that are based on procedures that were found to be effective for minimizing total tardiness in flowshops without family setups are described in this section. The first four procedures are neighborhood search procedures. Four of the procedures are variable greedy algorithms procedures and two procedures are genetic algorithms.

### Neighborhood Search Procedures

(Kim et al., 1996) developed two neighborhood searches for minimizing total tardiness in flowshops without family setups. One of the searches used an exchange operator to define the neighborhood and the other used an insert operator. Four neighborhood search procedures are presented in this section. The first two neighborhood searches use both of the neighborhood operators developed by (Kim et al., 1996). These procedures are referred to as NSA and NSB in this paper. The two procedures differ in how they obtain an initial solution. Two procedures for obtaining an initial sequence are proposed. The other two neighborhood search procedures use the insert and exchange operators on individual jobs and also use exchange and insert operators as well as a consolidate operator on batches of jobs to define neighborhoods. These procedures are referred to as BNSA and BNSB in this paper. In all of the procedures an initial sequence needs to be developed and then the sequence is improved by searching the neighborhood of the sequence. The procedures that generate initial solutions are referred to as the IS1 and IS2 procedures. The IS1 initial sequence is used to initialize the NSA and BNSA procedures and uses a method that is similar to that used in (Kim et al., 1996) with some modification to consider sequence-dependent family setups. The IS2 initial sequence is used to initialize the NSB and BNSB procedures and also uses a method similar to that of (Kim et al., 1996) but imposes the group technology assumption. The group technology assumption schedules all the jobs in a family as a single batch therefore this procedure develops two sets of sequences: jobs within each family and a family sequence.

Two stopping conditions are used in the procedures: 1) the searches fail to obtain an improvement or 2) a time limit is exceeded. The time limit is set to n*(M/2)*0.125 seconds. This

time limit was based on preliminary computational experiments and was set for all of the procedures described in this section.

### Variable Greedy Algorithm Based on (Framinan and Leisten, 2008)'s Variable Greedy Algorithm

A variable greedy algorithm combines features of both the variable neighborhood search (VNS) and the iterated greedy (IG) algorithm meta-heuristics. An IG algorithm generates solutions by using a destruction phase and a construction phase on a constructive heuristic. During the destruction phase k elements are removed from a solution to form a partial solution. During the construction phase a greedy constructive heuristic obtains a potential new solution by adding the previously removed elements. A VNS is used to overcome the problem of the IG stalling in a local optimum. When a VNS is used the size of the neighborhood that will be searched varies or, in the case of combining VNS with IG, the number of elements k that will be removed and then added back varies. Four variable greedy algorithms used in this research are based on (Framinan and Leisten**,** 2008)'s variable greedy algorithm. VG2A differs from VG1A only in how the initial solution is generated. The third and fourth variable greedy algorithms, referred to as VG1B and VG2B in this paper, include the batch insertion neighborhood search (BI) used in the BNS neighborhood searches in addition to the insertion-based local search after the slack-based greedy algorithm is performed.

### Genetic Algorithms Based on (Vallada and Ruiz, 2010)'s Genetic Algorithms

A genetic algorithm is a meta-heuristic search procedure that uses a multiple solution search technique. This approach has been found to quickly generate good solutions for a wide variety of scheduling problems. In a genetic algorithm, an initial population of chromosomes is first created, and then successive populations (or generations) of chromosomes are created using some methodology until a stopping condition is met. A chromosome corresponds to a solution for the problem. For this problem, solutions can be represented by a permutation sequence of the jobs. The jth gene in a chromosome corresponds to the job in the jth position of a sequence.

(Vallada and Ruiz, 2010) proposed three genetic algorithms for minimizing total tardiness in permutation flowshops and performed computational tests for these algorithms and other metaheuristics. They found that the genetic algorithm that used a crossover operator to generate individuals, called offspring, from two selected progenitors performed the best. This algorithm was modified to incorporate sequence-dependent family setups and is referred to as GA in this research. A second genetic algorithm that also incorporates a batch neighborhood search is referred to as GAB in this research.

## COMPUTATIONAL TESTS OF THE PROCEDURES

### Data and Performance Measures

The procedures described in the previous section were tested on problems of various sizes in terms of the number of families, number of machines, for four sets of distributions of due date

range and tightness, and three sets of distributions for family setup times. Each problem set consists of 10 problems. The problems within a problem set have the same number of families, same number of machines, family setup times are drawn from the same distribution, and the due dates for the jobs are generated using the same distribution. Twelve problem sets were generated for each combination of number of families (F) and number of machines (M). The number of families and machines included ranged from three to ten.

The number of jobs in a family was generated for each family in a problem using a uniform distribution over the integers 1 and 10. The processing times of the jobs for each machine were generated using a uniform distribution over the integers 1 and 100. The setup times on each machine for each family were randomly generated using a uniform distribution. Three setup distributions were used: (1) over the integers 1 and 50, (2) over the integers 1 and 100, and (3) over the integers 1 and 200. The due dates for the jobs were also randomly generated using a uniform distribution over the integers MS (1 – r – R/2) and MS (1 – r + R/2), where MS is an estimated minimum makespan found for the problem and R and r are two parameters called due date range and tardiness factors. Four sets of these parameters were used: R = 0.5 and r = 0.25 (set 1), R = 1.0 and r = 0.25 (set 2), R = 0.5 and r = 0.50 (set 3), R = 1.0 and r = 0.50 (set 4). The procedures were coded in Turbo Pascal and were tested on a Dell Inspiron 1525 1.6 GHz Lap Top computer for all of the problem sets.

The measure of performance used to evaluate the procedures for the test problems is the percentage reduction (% Red) of the total tardiness of the solution generated by each procedure compared to the total tardiness obtained for the problem if the IS2 procedure is used. % Red = $[(Z_{IS2} - Z_h)/ Z_{IS2}] * 100$, where $Z_{IS2}$ = the total tardiness for the problem if the IS2 procedure is used, and $Z_h$ = the total tardiness of the solutions generated by the heuristic procedures.

**Results of the Test**

The results show that the neighborhood searches that include neighborhoods based on batches of jobs and the genetic algorithms generally performed best. The BNSB procedure had the highest % RED for 12 problem sizes and setup distributions, BNSA, 10, GAB, five, and GA, four. Either the BNSA or BNSB procedure had the highest % RED for each of the four combinations of r and R. The variable greedy algorithms performed fairly well for the smaller problem sizes but their performance deteriorated as the problem sizes became larger.

The inclusion of the batch insertion search in the genetic algorithm and variable greedy algorithms did not result in a major improvement but including neighborhood searches based on batches of jobs did result in a major improvement for the neighborhood searches.

The results also show that as the setup time distributions become larger the percent improvement compared to the IS2 procedure generally declines for all of the procedures. In terms of the r and R parameters the percent improvement compared to the IS2 procedure is generally lower as the due dates become tighter (r = 0.5) and the range of due dates become smaller (R = 0.5) for all of the procedures.

Based on the results the Batch Neighborhood searches (BNS) and Genetic algorithms are recommended for the problem.

## CONCLUSION

In this paper ten procedures that were based on heuristics found to be effective for minimizing total tardiness in permutation flowshops were proposed for minimizing total tardiness for permutation flowshops with sequence-dependent family setups. These procedures included neighborhood searches, genetic algorithms and variable greedy algorithms.

The proposed procedures were tested on problems of various sizes in terms of the number of families included and the number of machines, three distributions of family setups, and four sets of distributions that determine the tightness of due dates and the range of due dates. The solutions generated were compared against solutions developed by one of the procedures used to generate an initial solution.

The results of the tests showed that the neighborhood searches that included neighborhoods based on batches of jobs and genetic algorithms were generally the most effective.

Possibilities for future research on the problem include the development of an exact procedure or good lower bounds for the problem.

## REFERENCES

Allahverdi, A., C. T. Ng, T. C. E. Cheng, M. Y. Kovalyov, (2008). A Survey of scheduling problems with setup times or costs, *European Journal of Operational Research*, 187, 985 – 1032.

Cheng, T. C. E., J. N. D. Gupta, and G Wang, (2000). A review of flowshop scheduling research with setup times, *Production and Operations Management*, 9 (3): 262 – 282.

Framinan, J. M., and R. Leisten, (2008). Total tardiness minimization in permutation flowshops: a simple approach based on a variable greedy algorithm, *International Journal of Production Research*, 46 (22): 6479 – 6498.

Hasija, S., and C. Rajendran, (2004). Scheduling in flowshops to minimize total tardiness of jobs, *International Journal of Production Research*, 42: 2289 – 2301.

Kim, Y., H. G. Lim, and M. W. Park, (1996). Search heuristics for a flowshop scheduling problem in a printed circuit board assembly process, *European Journal of Operational Research*, 91, 124 – 143.

Naderi, B., M. Zandieh, A. K. G. Balagh, and V. Roshanaei, (2009). An improved simulated annealing for hybrid flowshops with sequence-dependent setup and transportation times to

minimize total completion time and total tardiness, *Expert Systems with Applications*, 30, 9625 – 9633.

Parthasarathy, S., and C. Rajendran, (1997). A simulated annealing heuristic for scheduling to minimize mean weighted tardiness in a flowshop with sequence-dependent setup times of jobs – A case study. *Production Planning and Control*, 8: 475 – 483.

Ruiz, R., and T. Stutzle, (2008). An iterated greedy heuristic for the sequence dependent setup times flowshop problem with makepan and weighted tardiness objectives, *European Journal of Operational Research*, 187, 1143 – 1159.

Tavakkoli-Moghaddam, R. A. Rahimi-Vahed, and A. H. Mirzaei, (2007). A hybrid multi-objective immune algorithm for a flow shop scheduling problem with bi-objectives: weighted mean completion time and weighted mean tardiness, *Information Sciences*, 5072 – 5090.

Vallada, E., R. Ruiz, and G. Minella, (2008). Minimising total tardiness in the m-machine flowshop problem: A review and evaluation of heuristics and metaheuristics, *Computers & Operations Research,* 4: 1350.

Vallada, E., and R. Ruiz, (2010). Genetic algorithms with path relinking for the minimum tardiness permutation flowshop problem, *Omega*, 38: 57 – 67.

Ying, K. C., J. N. D. Gupta, S. W. Lin, and Z. J. Lee, (2010). Permutation and non-permutation schedules for the flowline manufacturing cell with sequence dependent family setups, *International Journal of Production Research*, 48 (8), 2169 – 2184.

Zhu, X., and W. E. Wilhelm, Scheduling and lot sizing with sequence-dependent setup: a literature review, (2006). *IIE Transactions*, 38, 987 – 1007.