A COMPARISON OF SCHEDULING APPROACHES FOR A MAKE-TO-ORDER ELECTRONICS MANUFACTURER

Susan K. Heath

Naval Postgraduate School
555 Dyer Road
Monterey, CA 93943, U.S.A.

Douglas J. Morrice

Red McCombs School of Business
The University of Texas at Austin
1 University Station, B6500
Austin, Texas 78712-0212, U.S.A.

ABSTRACT

In this paper, we compare two scheduling procedures designed to minimize setup costs for a make-to-order electronics manufacturing. While setup costs are important, quick response is highly valued by the manufacturer's customers and customer service is negatively impacted when jobs spend too much time in the system. To address this issue, we simulate the factory running with the schedules produced by these two procedures and compare the output based on the age of jobs remaining unprocessed at the end of one production shift. The simulation results show that the scheduling procedure that results in the lowest setup cost does not necessarily yield the best job age distribution.

1   INTRODUCTION

We consider a problem described by Loveland et al. (2007) and Monkman et al. (2007) in which a make-to-order electronics manufacturer needed to improve its production scheduling system in one of its factories. The problem was addressed by developing some scheduling procedures designed to minimize set-up costs. While these procedures improved set-up costs, there was some concern regarding their impact on the age of jobs in process. With a make-to-order business model, quick response is highly valued by the manufacturer's customers and customer service is negatively impacted when jobs spend too much time in the system.

In this paper, we compare two scheduling procedures developed to address the manufacturer's problem of set-up cost minimization. We simulate the factory for one production period (shift) using the schedules generated by each of the two procedures and compare the output based on the distribution of the age of jobs remaining in the process at the end of the shift.

The remainder of the paper is organized as follows. Section 2 provides the problem description. The schedul-

ing procedures are described in Section 3. The experimental design for the comparison of the two procedures is given in Section 4. Section 5 contains a description of the simulation model. The results from the simulation are provided in Section 6. Conclusions are provided in Section 7.

2   PROBLEM DESCRIPTION

This section and the next contain a brief description of the problem and the scheduling procedures. Loveland et al. (2007) and Monkman et al. (2007) provide more details.

This manufacturer's factory must balance between being highly flexible and responsive to changing, customized demand while maintaining a high rate of production and quick order turn-around. To achieve these goals, the factory was designed according to the lean manufacturing and just-in-time philosophies. Within tight space constraints, the factory must be able to manufacture a continually increasing variety of customized products.

The factory produces a wide variety of products that are grouped into product families to aid in the production planning process. A product family is defined by the chassis type (i.e., case) required by a product. Hence, the terms "chassis" and "product family" will be used interchangeably. Each chassis has a variety of components that can be assembled with it. We will refer to a product family's chassis and set of components collectively as the product family's parts. Each family uses some components that are not used by any other family and some components that are used by various other families. Similarly, some components are used by only one family and others, by several families.

The factory has multiple identical production lines located in parallel. Figure 1 depicts a rough schematic of one of these production lines. As shown, each production line has spaces designed to hold pallets of chassis and spaces designed to hold boxes of components. Each production line can hold L (≥ 1) different types of chassis

(one type of chassis per pallet) and C (≥ 1) different types of components (one type of component per box). Each chassis space on a line is called a *lane* and each component space is referred to as a *bin*. When the line is running, a worker selects one chassis from the L lanes for the next product to be built and sends the chassis to the workers down the line. The workers down the line select the correct components for that customer order from the bins and add them to the chassis. The production computer system keeps track of which chassis types are in which lanes and which components are in which bins, and signals the workers to select the correct chassis and components for each custom order. Therefore, each production line can produce any customized product from any subset of L families, in any sequence, without requiring any changes to the types of chassis that are in the lanes.
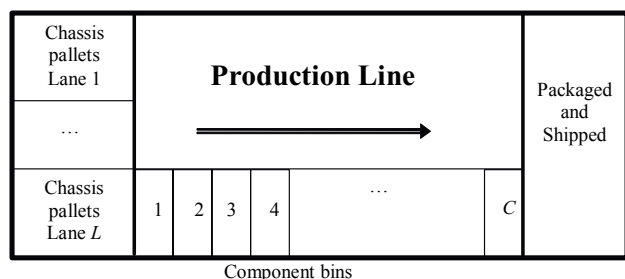

Figure 1: Layout of a single production line with L chassis lanes, and C component bins

A setup must take place when a line needs to produce products from a family whose chassis type is not in any of the lanes on the line. The setup entails removing the pallet for one chassis type from a lane and placing the pallet of the chassis type required for the next order in that lane. Components must also be changed as well. Those that are not needed for the other chassis types remaining in the other lanes, or for the new chassis type being placed in a lane, must be removed from the bins. Then, the components that are required for the new chassis type that are not already on the line must be placed in the bins. And, finally, the production computer must be updated with the new information on which parts are in which spaces.

The costs for this type of setup are sequence-dependent, but what is not immediately apparent is that these costs depend on the current *subset* of L families on the line and the new *subset* of L families on the line after the setup. Here, the total number of parts that will need to be changed during a setup equals the number of parts currently on the line that will no longer be needed by the new subset of L families plus the number of parts that are not currently on the line that will be needed by the new subset of L families. We use the number of parts that need to be changed during a setup as the setup cost.

3   SCHEDULING PROCEDURES

The two scheduling procedures are heuristics that take a "divide and conquer" approach to what would otherwise be an intractable problem. In the spirit of Pinto and Grossmann (1998), we developed procedures that address each of the three major scheduling steps separately: assign tasks to equipment, sequence the tasks, and schedule the exact time that each task will occur within the sequence. In our problem, the "tasks" are product families and the "equipment" is the production lines. The timing of a task is actually the time that the parts for a given product family are placed on the assigned production line.

Both procedures use the same mixed integer programming model for the assignment step. A solution to this model assigns demand and components for all product families to lines, ensures that enough demand is assigned to every line in order to keep it fully utilized, minimizes the number of setups, and minimizes the maximum number of components on any line. Two additional constraints are included in this assignment model for managerial purposes because together they allow for some shifting of demand across lines without requiring a different chassis-line assignment. These constraints ensure that each production line has chassis with very high demands (high runners) assigned to it, and every high runner is assigned to more than one line. One further constraint limits the number of families assigned to a single line to a maximum established by the company.

The line assignment problem was programmed into GAMS and solved by calling Cplex 9.0. Practical problems are quite large and difficult to solve to optimality within a time limit that management finds acceptable. Therefore, the computations were halted when the optimality gap reached 0.5% which was achieved within 3 minutes for all cases considered. See Monkman et al. (2007) for model formulation and solution details.

The two procedures differ significantly in the next two steps. The first procedure, referred to as Procedure 1, relies on the simplifying assumption that set-up costs are sequence independent. While this assumption is violated in practice, it is an assumption the manufacturer is willing to make in order to have a procedure that can produce approximate solutions quickly. In contrast, Procedure 2 accommodates sequence dependent set-up costs. Consequently, steps 2 and 3 of Procedure 2 are more sophisticated and accurate but harder to solve.

For Procedure 1, the sequencing step reduces to another assignment problem in which product families are assigned to lanes on a line and then sequenced in each lane using a simple heuristic. The heuristic tries to:

1.   balance demand across lanes,

2. ensure that a chassis is assigned to a particular lane and put first in the sequence if it is left on that lane at the end of the previous shift,
3. assign each chassis to exactly one lane,
4. make sure each lane has at least one chassis, and
5. sequence the remaining families in each lane based on the level of demand, from highest demand chassis to lowest demand chassis.

For the third step in Procedure 1, each lane's available production time for the production period is allotted to each chassis type in proportion to its expected percentage of the total demand assigned to that lane. This allotment determines the setup times within each lane. Then, if any of the lane setup times overlap the lane setup times are shifted slightly so that no more than one family is being changed at a time. The heuristic also takes into account planned downtimes due to breaks or scheduled maintenance by decreasing each lane's available production time accordingly.

Procedure 2 does not rely on the assumption that setup costs are sequence independent. With traditional sequence-dependant setup costs, the sequencing problem can usually be modeled as a traveling salesman problem (TSP) where each task is represented by a node in the graph, and each arc between two nodes represents the setup cost when switching from one task to the other task. To obtain a solution where all tasks are completed, each node must be visited. Since our setup costs depend on the subsets of L families on the line before and after the setup, we needed to create a graph for our problem where each node represents a possible subset of L families, with the arcs representing the setups costs of changing between these subsets. With this formulation, it is no longer necessary to visit every node, but rather to make sure enough nodes are included in the solution so that each family will be on the line for a sufficient amount of time during the shift. With this formulation, the sequencing step in Procedure 2 can be formulated as a traveling salesman subtour problem (TSSP) which is sometimes referred to as a traveling salesman subset-tour problem (Mittenthal and Noon, 1992). Since the TSSP is an NP-hard problem (it is a special case of a TSP), it is difficult to solve practical problems to optimality. A greedy randomized adaptive search procedure (GRASP) (Feo and Bard, 1989) is developed and applied to quickly find a good, but not necessarily optimal, solution to this sequencing TSSP.

The third step in Procedure 2 attempts to allot each lane's available production time to each chassis type in proportion to its expected percentage of the total demand assigned to that lane. However, this is more complicated than in Procedure 1 because families are sequenced on each line in subsets in step 2 of Procedure 2. Consequently, the allotment of time to families is adjusted by changing when each node (each subset of families) is

scheduled to be on the line, with a setup duration between each node's end time and the next node's start time. An integer programming model was formulated for this problem, programmed into GAMS, and solved to optimality using Cplex 9.0.

4 COMPARISON EXPERIMENTAL DESIGN

While Procedure 2 is harder to solve than Procedure 1, it does yield significantly better solutions in terms of setup costs. Monkman et al. (2007) report improvements of roughly 18%, on average, and 49% in the best case.

The purpose of this study is to compare these two procedures on another dimension: age of jobs remaining in the system after running one production shift using the schedules generated by the procedures. The comparison is done using simulation.

One might be tempted to conclude that Procedure 2 will outperform Procedure 1 on this second dimension as well since the former is more efficient at setups. But this is not so clear because the two objectives are quite different and not necessarily complimentary. Additionally, the third step of Procedure 2 is less flexible than Procedure 1's third step because in Procedure 2, time on the production line is allotted by subsets of families rather than by individual family. This means that the decision to allot more or less time to one family on a line is restricted not only by the production needs of the other families in the same lane (as in Procedure 1) but also by the needs of the other families in the node, and, as a consequence, all the other families on the same production line. It seems plausible that this loss in flexibility might increase the time jobs spend in the system.

The data used for the simulation model were gathered from the manufacturer's factory over a two week time period. Each day is subdivided into production shifts and each shift is broken down into production run time periods. These data contained detailed job information for each production run including the family type, the order date for each job, and the date that each job becomes available for factory processing. Note that the order date and the factory processing dates can differ if a job gets delayed in order processing. Delays can occur due to many factors such as credit checks, parts shortages, and unexpectedly high demand. From these data, we were able to estimate probability distributions for the age and number of jobs arriving to the factory for each family in each production period using Palisade Decision Tool's BestFit, Version 4.5 (www.palisade.com). Distributions for the number and age of jobs in the system at the beginning of the simulation (backlog) were estimated from the data collected and from expert opinion.

In order to control for variation and isolate the impact of the different scheduling approaches on the age of jobs in the system, we used common random numbers. To help

synchronize the random number streams across the different simulations, we used separate random number streams for each probability distribution in the simulation model (Law and Kelton, 2000, page 588).

The comparison experiment is run in the following manner:

1. The simulation model randomly generates an initial number of jobs for each family (backlog) and the age of each job in backlog.
2. The generated backlog plus the expected number of jobs for each family over a production shift is input into the scheduling heuristics to produce the schedules.
3. Each schedule is read back into a separate run of the simulation and one production shift of the factory is simulated.
4. The job age results for the jobs in backlog at the end of the shift for each scheduling procedure's simulation run are dumped to a data file for analysis in Excel.
5. Steps 3 through 4 are replicated, with only newly arriving demand changing between replications, to get statistics for the job age distributions for the schedule produced by each scheduling procedure.
6. The job age distribution statistics are compared to determine if one procedure is performing significantly better than the other.
7. Steps 1 through 6 are replicated to get statistics for multiple schedules.

5 SIMULATION MODEL DESCRIPTION

The simulation model was developed in Rockwell Automation's Arena simulation software (www.arenasimulation.com/default.asp). It has four main sections: creation of backlogged and newly arriving jobs (Figure 2); the production line (Figure 3); family changeovers on production lines (Figure 4); and output (not shown).

The logic for creating backlogged and newly arriving jobs in Figure 2 for product family M is representative of the logic used for all families. Backlogged jobs are created at the beginning of the simulation to represent jobs that have been held over from previous production shifts. All other jobs are created during the simulation runs. This simulates new customer order information arriving to the factory over the course of the production shift. Both backlogged and newly arriving jobs are assigned attributes including family name and a priority based on age (oldest gets the highest priority so it can be processed first when its family is on the line).

The scheduling procedures generate schedules that specify the percentage of jobs from a family that are assigned to each production line. Based on constraints set by the management, each family can be assigned to either one line or two lines. Figure 2 depicts a family that has been assigned to two lines. Hence, when jobs reach the "Decide" module, the assigned percentage are routed to one line and the rest are routed to the other. Jobs for families that are only assigned to one line get routed directly to their assigned line without a decide module.
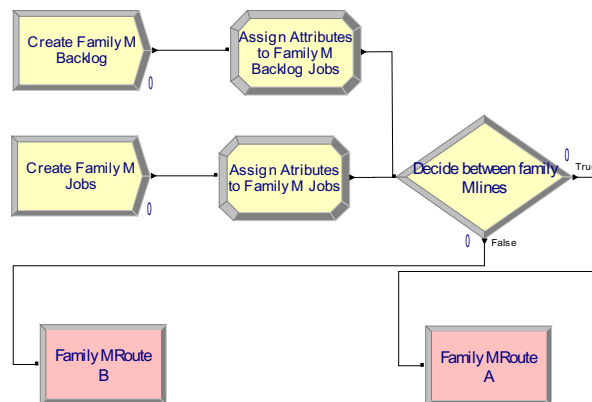

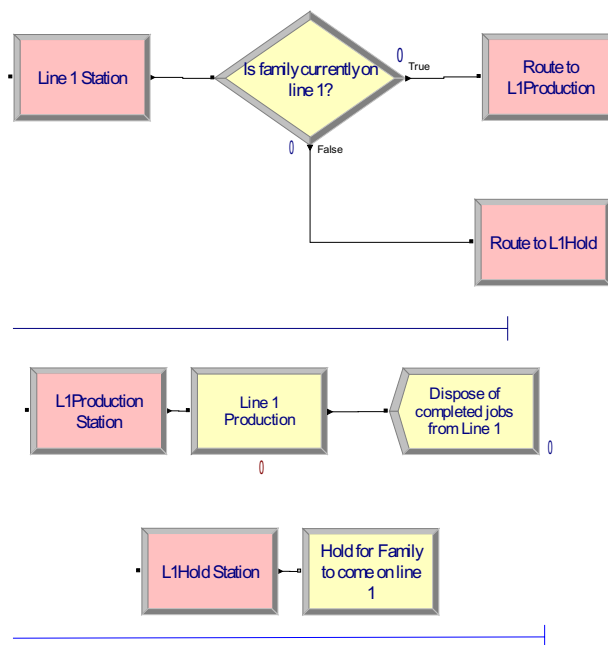Figure 2: Example of Arena logic for creating backlogged and newly arriving jobs

Figure 3: Example of Arena logic for production lines

Figure 3 contains the logic for one of the production lines (i.e. Line 1). All other production lines have the same logic. When a job arrives at the production line it is queued up for production if its product family is currently scheduled on one of the lanes on the line. Otherwise, the

job is routed to a holding queue where it is held until the family gets scheduled on the line. Once in production, an job undergoes a production time delay after which it is disposed. Due to the fact that the production is high volume and fairly repetitive, a constant production delay is used.

Figure 4 contains the logic for product family changeovers on a line. The logic is shown for Production Line 1 but again the logic is the same for other lines. The family changeover logic is arguably the most complex in this simulation model.

The create node initiates a cycle in which information for the next changeover is read in from a file produced by the scheduling procedure. After the simulation clock advances to the next changeover time, one family is removed from the line and another family is added. The removal entails determining the lane in which the family is currently scheduled and removing it from this lane. Then the production line queue is searched for any jobs corresponding to this family. When a job for this family is found in the production line queue the job is removed and routed to that line's holding queue and the search is repeated until no more jobs from that family are in the production line queue.

After a delay for setup time, the product family being added is assigned to the empty lane. Then the holding queue is searched for any jobs in this family. When a job for this family is found in the holding queue the job is removed and routed to that line's production queue and the search is repeated until no more jobs from that family are in that line's holding queue. This completes the scheduled

changeover and the cycle repeats, returning to read in the next changeover information.

The last part of the model (not shown) is invoked at the end of the simulation of one production shift. It searches through all the production line and holding queues and dumps the family type and age for each job remaining in the system to a data file. The data file is imported into Excel for the analysis contained in the next section.

6 SIMULATION EXPERIMENT

The experimental steps described at the end of Section 4 are used to generate data for comparison. Table 1 contains statistics on the age of the jobs remaining at the end of the simulated production shift for fifteen replications using the same set of initial conditions (same backlog and same schedule). Newly arriving demand is randomly generated for each replication. On each replication, both procedures face exactly the same newly arriving demand because common random number streams are used. Therefore, the replicates are paired and the difference between age of jobs distribution statistics (Procedure 1 - Procedure 2) form the basis of the analysis. In its second column, Table 1 contains the differences for the mean, standard deviation, skewness, count, and several percentiles averaged over the fifteen replications. The third column in Table 1 provides results from a hypothesis test with null hypothesis that the average difference is zero and alternative hypothesis that the average difference is less than zero.
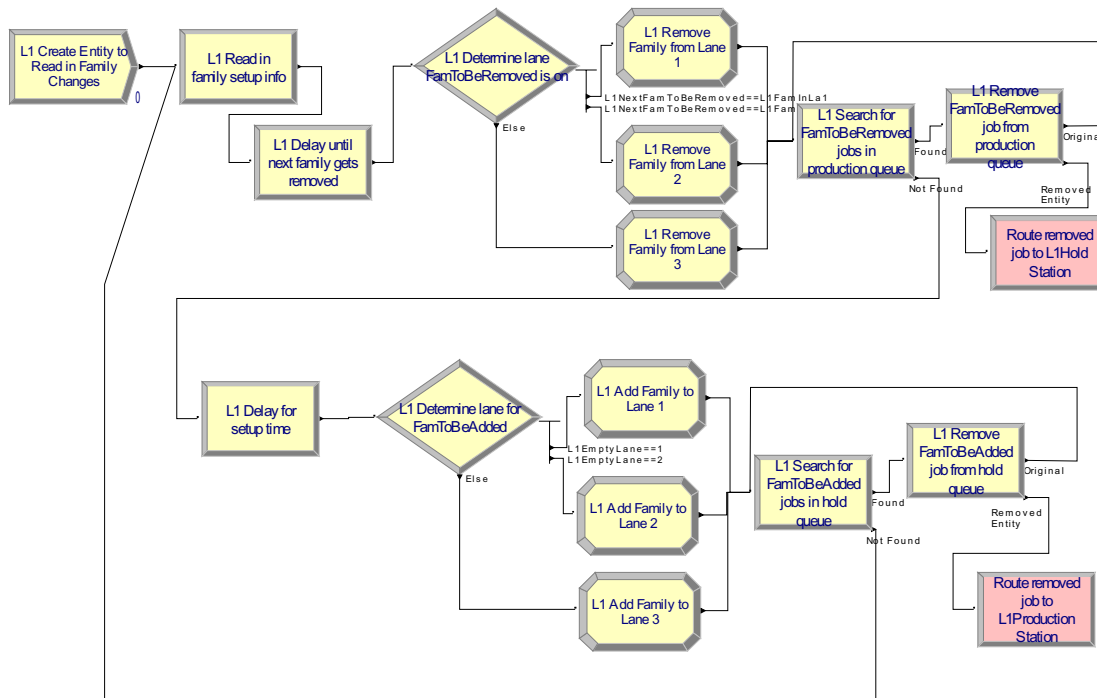

Figure 4: Example of Arena logic for product family changeovers on a line

On average, job ages are significantly higher for Procedure 2. Comparison of the percentiles of the job age distributions helps explain why this is so and reveals important information about the performance of the two procedures. Up to the 80th percentile the two distributions have very similar percentile values. At the 90th percentile and above, the percentiles for Procedure 2 significantly larger than for Procedure 1. In other words, the upper tail of the job age distribution tends to be longer for Procedure 2 which explains why the mean of its distribution is higher. These results indicate that most jobs (80+ %) have comparable ages under both procedures but the oldest jobs are significantly older for Procedure 2. In other words, certain old jobs that are processed using Procedure 1's schedule during the single production shift do not get processed under Procedure 2's schedule. Additionally, based on the count statistic, Procedure 2 leaves significantly more jobs in the system at the end of the shift on average.

Table 1: Average Differences Between Age of Jobs Distribution Statistics for the Scheduling Procedures (Procedure 1 – Procedure 2) Over 15 Replications of a Single Schedule

Statistic	Average Difference (Procedure 1 – Procedure 2)	Hypothesis Test (Lower Tail Test)
Mean	-105.86	Reject at 1%
Std. Dev.	-1063.09	Reject at 1%
Skewness	-8.03	Not significant
Median	0.58	Not significant
Count	-219.80	Reject at 1%
1st Quartile	-1.55	Reject at 5%
3rd Quartile	14.66	Not significant
1.00%	0.00	Not significant
2.50%	-0.05	Not significant
5.00%	0.06	Not significant
10.00%	-1.46	Reject at 5%
20.00%	-0.28	Not significant
80.00%	13.57	Not significant
90.00%	-12.27	Reject at 1%
95.00%	-288.57	Reject at 1%
97.50%	-1187.67	Reject at 1%
99.00%	-2440.73	Reject at 1%

In order to check these results further, the experiment just described was repeated for two more sets of initial conditions (i.e., two more schedules and sets of backlogged jobs). Table 2 provides the combined results that are based on fifteen replications for each of three different schedules (45 replications total). While the same patterns hold, the results provide even stronger evidence that Procedure 2 is not clearing older jobs out as well as Procedure 1.

The results in Tables 1 and 2 are probably best explained by the fact that step 3 in Procedure 2 is less flexible than the corresponding step in Procedure 1 (see the discussion in Section 4). However, while Procedure 2 has worse performance on age of jobs, it has significantly better performance on setup costs are measured by the number of part swaps required in the given schedule. Table 3 illustrates that for the three schedules considered, Procedure 2 yields roughly a 40 percent improvement over Procedure 1.

Table 2: Average Differences Between Age of Jobs Distribution Statistics for the Scheduling Procedures (Procedure 1 – Procedure 2) Over 3 Schedules of 15 Replications Each

Statistic	Average Difference (Procedure 1 – Procedure 2)	Hypothesis Test (Lower Tail Test)
Mean	-54.04	Reject at 1%
Std. Dev.	-578.67	Reject at 1%
Skewness	-5.41	Not significant
Median	-2.36	Reject at 5%
Count	-277.11	Reject at 1%
1st Quartile	0.90	Not significant
3rd Quartile	-7.07	Reject at 5%
1.00%	0.01	Not significant
2.50%	0.09	Not significant
5.00%	0.68	Not significant
10.00%	1.28	Not significant
20.00%	1.35	Not significant
80.00%	-16.64	Reject at 1%
90.00%	-38.72	Reject at 1%
95.00%	-125.03	Reject at 1%
97.50%	-424.93	Reject at 1%
99.00%	-967.24	Reject at 1%

Table 3: Best Solution Setup Costs for the Scheduling Procedures on the Three Schedules Considered

Schedule	Procedure 1 Setup Costs	Procedure 2 Setup Costs
1	652	374
2	962	560
3	816	488

One last statistic of interest is the utilization of the production lines under each scheduling procedure. Table 4 contains the average utilization for each line under each procedure over 45 replications (three schedules of fifteen replications each). The results are mixed. On average,

Procedure 1 has higher line utilization on lines 2, 3, and 5 and Procedure 2 has higher utilization on lines 1 and 6. All these results are statistically significant at at least the five percent level according to a paired t-test. There is no difference in utilization for Line 4 since it is fully utilized on both procedures.

These utilization results are surprising. Each of the scheduling procedures is designed to keep the production lines busy 100% of the time. In addition, both of the procedures assign the same amount of demand to each production line. It turns out that the scheduling procedures actually allow the possibility of a line receiving less than the demand required to keep it 100% utilized if an unexpectedly low amount of newly arriving demand actually arrives during the shift. Also, even though a line may have enough total demand assigned to it, some jobs might be sitting in the holding queue while the line idles because that family is not scheduled to be on the line. However, in practice the factory is able to make final adjustments to the schedule before they implement it so they can shift some demand from a line with more than enough demand to a line that has much less demand to prevent these idling issues. Adjusting the scheduling procedures to prevent this situation one task in future research.

Table 4: Average Utilization for Each Line on Each Procedure over Three Schedules of 15 Replications Each

Line	Procedure 1 Average Utilization	Procedure 2 Average Utilization
1	0.964	0.966
2	0.987	0.892
3	0.978	0.956
4	1.000	1.000
5	0.970	0.930
6	0.976	0.978

7 CONCLUSION

In this paper, we have used simulation to compare the performance of two scheduling procedures designed to minimize setup costs for a make-to-order electronics manufacturer. The simulation results indicate that the procedure which yields the best solution for setup costs does not necessarily perform the best with regard to the age of jobs remaining in the system after one production shift.

Future work on this problem includes looking for ways to improve Procedure 2's performance on the age of jobs remaining metric. One might approach this by extending the objective function of the optimization problem to include a loss function that penalizes the aging of jobs. However, since the current model is already quite difficult to solve to optimality, this might not be all that practical.

In addition, one might simulate more than one production shift to understand the how jobs age over multiple production shifts. This might be extended further to develop a combined optimization – simulation procedure in which backlog at the end of each shift is used to generate a new schedule and help control job aging as the simulation progresses.

REFERENCES

Feo, T. A. and J. F. Bard. 1989. Flight scheduling and maintenance base planning. *Management Science* 35(12): 1415-1432..

Law, A. M., and W. D. Kelton. 2000. *Simulation modeling & analysis*. 3rd ed. New York: McGraw-Hill, Inc.

Loveland, J. L., S. K Monkman., and D. J.Morrice. 2007. Dell uses new production scheduling heuristics to accommodate increased product variety. *Interfaces*, 37(3): 209-219.

Mittenthal, J. and C. E. Noon. 1992 An insert/delete heuristic for the traveling salesman subset-tour problem with one additional constraint. *Journal of the Operational Research Society* 33(3): 277-283.

Monkman, S. K., D. J. Morrice, and J. F. Bard. 2007. A production scheduling heuristic for an electronics manufacturer with sequence dependent setup costs. To appear in *European Journal of Operational Research*.

Pinto, J. M. and I. E. Grossmann. 1998. Assignment and sequencing models for the scheduling of process systems. *Annals of Operations Research* 81:433-466.

AUTHOR BIOGRAPHIES

SUSAN K. HEATH is an assistant professor of logistics and operations management in the Graduate School of Business and Public Policy at the Naval Postgraduate School. She received her Ph.D. in management science and information systems from the University of Texas at Austin in 2006. She also has a Master of Engineering degree in operations research and industrial engineering and a Bachelor of Science in psychology, both from Cornell University. From 1997 to 2000 she worked as a senior business analyst in information systems for Kraft Foods, Inc. She has published in Interfaces, has a forthcoming publication in the European Journal of Operational Research, and has been a guest speaker at several conferences and universities. Her email address is <skheath@nps.edu>.

DOUGLAS J. MORRICE is a Professor in Operations Management at The University of Texas at Austin. He has ORIE Ph.D. from Cornell University. His research interests include simulation design, modeling, and analysis. Dr. Morrice was Co-Editor of the *Proceedings of the 1996*

Winter Simulation Conference, and 2003 Winter Simulation Conference Program Chair. He is currently serving as a representative for the INFORMS Simulation Society on the Winter Simulation Conference Board of Directors. His email address is <morrice@mail.utexas.edu>.