Available online at www.sciencedirect.com
SCIENCE DIRECT®
Int. J. Production Economics 96 (2005) 273–283
international journal of production economics
www.elsevier.com/locate/dsw

A beam search heuristic method for mixed-model scheduling with setups

P.R. McMullen[a,*], Peter Tarasewich[b]

[a]Babcock Graduate School of Management, Wake Forest University, Winston-Salem, NC 27109, USA
[b]Northeastern University, College of Computer and Information Science, 360 Huntington Avenue, Boston, MA 02115, USA

Received 1 May 2003; accepted 1 December 2003
Available online 17 July 2004

Abstract

Mixed-model scheduling involves determining a production sequence for multiple products along a single assembly line. While the goal of using this approach is often determining a schedule which keeps the usage of parts as constant as possible, there is an implicit but often unrealistic assumption that setup times are negligible. When setup times are assumed to be significant, the sequencing decision becomes a multi-objective problem which needs to minimize both the usage rate and the number of setups.

However, the objectives of a low usage rate and a minimum number of setups are frequently in opposition with one another. To address this situation, this research presents an efficient frontier approach to support sequencing decisions that are made as a result of the compromise necessitated by these two conflicting goals. A beam-search heuristic is used to effectively generate the efficient frontiers needed to solve the problem.

Keywords: Just-in-time systems; Beam search; Heuristics; Dual-objective decision-making

1. Introduction

Just-in-time (JIT) is a management philosophy that uses a set of integrated activities to achieve manufacturing flexibility with minimal inventories. Successful implementation of JIT production systems can help organizations achieve competitive priorities such as low cost and consistent quality (Gilbert, 1990; Huson and Nanda, 1995). Specific management practices associated with JIT manufacturing systems vary, but include total quality control, uniform workload, multifunction employees, and reduced setup times (White and Ruch, 1990). Overall, adopters of JIT have shown significant improvements in performance (White, 1993).

One manufacturing problem that is often associated with JIT practices is mixed-model

*Corresponding author. Tel: 336-758-4574.
E-mail address: patrick.mcmullen@mba.wfu.edu (P.R. McMullen).

0925-5273/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.ijpe.2003.12.010

scheduling. Solving this problem involves determining a production sequence for multiple products along a single assembly line. Manufacturers will use this type of scheduling technique to meet diversified customer demands while maintaining minimal product inventories. Because JIT systems are concerned with having the right parts at the right place at the right time, the goal of a company using this approach is often determining a schedule which keeps the usage of every part in the assembly line as constant as possible.

But the implicit assumption in the mixed-model sequencing problem is that setup times between the different products on the assembly line are negligible. A setup is required each time two consecutive items in the production sequence are different. While it is acknowledged that small setup times are an essential ingredient for JIT success, it is more realistic in certain cases to assume small but non-negligible setup times. This research addresses the mixed-model assembly line problem with significant setup times.

When setup times are taken into account, the goal of the problem changes. It becomes sequencing products as evenly as possible while minimizing the number of setups that occur when switching between different products. A good sequence of products should not only have an acceptable level of product inter-mixing, but also an acceptable number of required set-ups. In this situation, the sequencing decision becomes a multi-objective problem, minimizing both usage rate and number of setups.

The problem is challenging because the goals of a low usage rate and a minimum number of setups are frequently in opposition with one another (McMullen and Frazier, 2000). A manager of a production line must determine desirable sequences with respect between setups and usage rates. To lessen the difficulties in assessing these tradeoffs, this research uses an efficient frontier approach to support the decision that is made as a result of the compromise between the two conflicting goals.

This paper proceeds as follows. First, a description of the mixed-model scheduling problem with setups is given. Next, an efficient frontier approach to aid in the determination of effective solutions to the problem is presented. A beam-search heuristic is then used to effectively generate the efficient frontiers needed to solve the problem. The paper concludes with a summary of the work and future research directions.

2. Problem definition

An assembly line producing more than one product at a given time in an intermixed fashion is known as a mixed-model line. Higher performance with this type of production line can be achieved through optimized product flow. Often a goal in scheduling production on the mixed-model line is keeping the usage of parts for the different products being produced as level as possible. Constant use of parts allows easier implementation of a JIT manufacturing environment due to lower variations in production quantities and work-in-progress inventories. This problem has been widely studied, first by Monden (1983), and more recently by Miltenburg and Sinnamon (1989, 1992, 1995), Sumichrast et al. (1992), Bolat (1994), Tamura et al. (1999), Bard et al. (1992), Inman and Bulfin (1991), Xiaobo and Ohno (1994), and Xiaobo et al. (1999). As another source for the curious reader, Yano and Bolat (1989), and Ghosh and Gagnon (1989) offer research pertaining to the planning and scheduling of assembly systems.

The mixed-model sequencing problem, however, implicitly assumes negligible setup times between the different products on the line. While minimal setup times are a key to JIT success, it is more realistic in certain cases (e.g., automobile assembly) to assume significant setup times. Here, the goal of the problem changes to sequencing products as evenly as possible while minimizing the number of setups that occur—a multi-objective problem, where the objectives are frequently in opposition with each another (McMullen and Frazier, 2000).

Consider the situation where four units of item A need processing, along with two units of item B and one unit of item C. If minimization of setups were desired, one possible sequence would be: AAAABBC. This sequence would result in the minimum of three setups (a required setup is

assumed here for A at the start of the processing). Unfortunately, this sequence does not provide evenness of the material usage rate (or desirable intermixing). The following sequence does provide more stability of the material usage rate: ABACABA. Unfortunately, this sequence requires the maximum of seven setups. These two simple examples illustrate the tradeoff between required setups and stability of material usage rate—a factor which complicates decision-making.

A mathematical formulation for the mixed-model scheduling problem with setups is now presented. First, the following parameters are defined, and it is detailed as to whether these parameters are determined by the system (endogenous) or pre-determined (exogenous):

U	usage rate of a production sequence (endogenous)
S	number of setups in a production sequence (endogenous)
U_S	usage rate of production sequence associated with S setups (endogenous)
a	number of unique products to be produced (exogenous)
d_i	demand for product i, $i = 1, 2, \ldots, a$ (exogenous)
D_T	total number of units for all products or total demand—also represents number of positions in sequence (exogenous)
s_k	1 if setup required; 0 otherwise (endogenous)
$x_{i,k}$	total number of units of product i produced over stages 1 to k, where $k = 1, 2, \ldots, D_T$ (endogenous)

The problem has two objective functions. The first is the minimization of the number of required setups. The number of setups (S) in a production sequence is

$$\text{Minimize}: \quad S = 1 + \sum_{k=2}^{D_T} s_k, \tag{1}$$

where k is the index of the position in the sequence. If the item in position k is different from the item in the previous position (position $k-1$), then a setup is required. Mathematically, this is as follows: $s_k = 1$ is setup is required in position k; Otherwise $s_k = 0$. The following assumptions are made regarding setups:

- An initial setup is required regardless of sequence, which is why the index k starts at 2.
- The time required to perform a setup is assumed to be sequence-independent, so the setup time for an item on a machine is not determined by the item which precedes it.
- The number of setups required represents the required setup time. Therefore, the total setup time is directly proportional to the total number of setups.

The second objective is to optimize the stability of raw material usage rates. Monden (1983) was the first to formally recognize this important objective as it relates to JIT sequencing. Miltenburg (1989) developed a metric to measure this stability of parts usage. This metric is referred to as the Usage Rate (U), and is defined as follows:

$$\text{Minimize}: \quad U = \sum_{k=1}^{D_T} \sum_{i=1}^{a} \left(x_{i,k} - k \frac{d_i}{D_T} \right)^2. \tag{2}$$

Stability of parts usage will be achieved through minimization of the above usage rate. Thus, there are two objectives of interest here for JIT production sequences: minimization of both setups and usage rate.

Both objectives are subject to the following constraints, guaranteeing the pre-specified product-mix:

$$x_{i,D_T} = d_i, \quad \text{for } i = 1, 2, \ldots, a. \tag{3}$$

3. An efficient frontier approach

Past research has demonstrated that there is an inherent tradeoff between usage rate and the number of setups McMullen and Frazier (2000). As a result, the two objectives, which oppose each other, must be simultaneously optimized. One way to do this is to construct a composite objective function, which could search for desirable combinations of number of setups *and* usage rates.

Construction of a composite objective function involves weighting both components of the objective function, which requires the decision-maker to determine weighting schemes. Determination of weighting schemes can be problematic, due to the fact that the user is dealing with two metrics measured in different units—which can have confounding effects on the composite objective function value.

To avoid the potentially dangerous issue of weighting schemes, an efficient frontier approach is used here to obtain sequences having desirable combinations of numbers of required setups and usage rates. This approach also presents the decision-maker with a graphical tool to aid in the process of evaluating trade-offs and selecting the final production sequence to be implemented. This type of problem lends itself well to an efficient frontier approach because there are exactly two objectives (number of setups and usage rate), with one being a discrete measure (number of setups) and the other a continuous measure (usage rate). This means that the decision-maker could strive to obtain minimal values of the continuous measure (usage) for each value of the discrete measure (number of setups). If the number of setups were continuous and not discrete, the approach described above would not be possible. An efficient frontier is a collection of points where one axis typically presents a discrete variable, while the other presents an optimal value of another variable at each unique level of the discrete variable. An efficient frontier approach has become a popular means of addressing multiple objective optimization problems.

The example in Fig. 1 shows how the number of setups and usage rate could be translated into an efficient frontier. Point 1 is on the frontier, while Point 2 is not. Both points have the same number of setups, but the sequence represented by Point 2 has a higher usage rate than the sequence represented by Point 1. In short, the sequence represented by Point 2 is dominated by the sequence represented by Point 1. Similarly, the sequence represented by Point 4 is dominated by the sequence represented by Point 3 for the same reasons, although the number of setups here is different than for the sequences represented by Points 1 and 2.

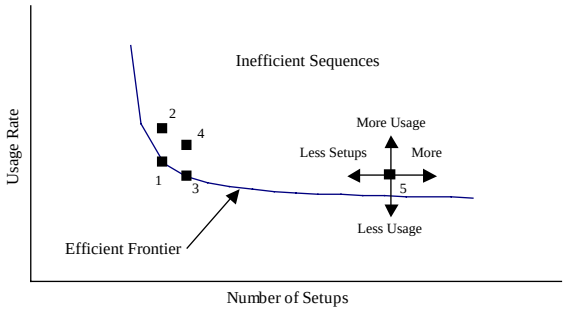

Fig. 1. Illustration of efficient frontier.

Any sequence having a combination of number of setups and usage rate that is on the efficient frontier is considered dominant. Any sequence "northeast" of the frontier is considered dominated by the sequences on the frontier.

3.1. Determining the best mixed-model schedule using the efficient frontier

Use of an efficient frontier affords managers and other decision-makers with the ability to find sequences providing acceptable levels of both number of setups and usage rates. The way this is done, is rather straightforward: the decision-maker first examines the efficient frontier that has been generated for the sequencing problem of interest and chooses the maximum number of setups that they can be comfortable with. From there, they find the subsequent sequence resulting in the minimum usage rate as presented above.

The continuous nature of the efficient frontier also provides the ability to perform some sensitivity analyses. For example, a decision-maker could find a "neighboring" point along the frontier, where they can exploit a sequence's property of the usage rate/number of setups tradeoff to find the most appropriate sequence for their needs. Point 5 in Fig. 1 illustrates a possible starting point for such a sensitivity analysis.

This graphical method of displaying a set of efficient solutions provides a simple yet effective way for managers to quickly evaluate the tradeoffs involved in choosing a manufacturing sequence. It

helps to prevent potentially, poor decisions that can result from a single solution calculated from user-determined weighting values.

Given that this research effort is concerned with finding the minimal usage rate for each associated number of required setups, the objective function can be presented in traditional mathematical programming formatting as follows:

$$\text{Minimize}: \quad U_S = \sum_{k=1}^{D_T} \sum_{i=1}^{a} (x_{i,k} - k d_i / D_T)^2, \quad \text{for all unique values of } S \tag{4}$$

$$\text{subject to}: \quad x_{i,D_T} = d_i, \ \text{for } i = 2, 1, \ldots, a. \tag{5}$$

4. Constructing the efficient frontier

Construction of the efficient frontier for any list of items requiring sequencing requires the decision-maker to address the fact that problems of this type are combinatorial—small increases in problem size result in large increases in computational resources needed to find optimal solutions. The number of unique sequences when there are d_i units demanded for each of the a different items is expressed as follows:

$$\text{Unique sequences} = \frac{\left(\sum_{i=1}^{a} d_i\right)!}{\prod_{i=1}^{a} (d_i!)}. \tag{6}$$

Because these types of sequencing problems have so many possible permutations, traditional optimization techniques such as linear or integer programming are not practical approaches. As a result, search heuristics provide an opportunity to find desirable solutions with a reasonable expenditure of computational effort.

4.1. A beam search heuristic

Beam search is a heuristic search procedure that can be used to effectively address the problem of interest here. It is a modification of a breadth-first search (Norvig, 1992). Since its inception, beam search has been used for a variety of job-shop scheduling approaches (Fox, 1983; Ow and Morton, 1988; Ow and Morton, 1989; Ow and Smith, 1988). De et al. (1992) used a beam search approach to construct an efficient frontier exploiting a weighted linear combination of mean and variance of job completion time on a single machine. Nair et al. (1995) used beam search to design product lines with the objective of optimizing the mutual satisfaction of buyers and sellers.

All permutations of the problem addressed here can be represented via a search tree—the leaves at the bottom level of the tree represent all possible sequences. When branching from a node to its lower-level nodes, beam search only permits branching on the b most promising lower-level nodes (the other nodes are permanently pruned). The parameter b is referred to as the beam width, and the decision-maker chooses its value. Beam search results in less branching than the breadth-first search, which subsequently results in less computational effort. Another parameter chosen by the decision-maker is the pruning depth, depth, which is the level in the search tree where actual pruning commences. Full branching takes place at all levels of the search tree above depth. Giving the decision-maker control over the value of depth provides them with control as to how much branching will occur prior to pruning. For both beam width b and depth, larger values result in more sequences being evaluated—this also results in the requirement of more computational resources.

The search tree in Fig. 2 shows all permutations of the simple sequencing problem where there are three different products ($a = 3$). There are two units of item A demanded ($d_1 = 2$), and one unit each of items B and C ($d_2 = 1$, $d_3 = 1$), which results in 12 possible sequences:

$$\frac{(2+1+1)!}{(2!)(1!)(1!)} = \frac{24}{2} = 12.$$

At each level of the search tree, the items to the left of the vertical bracket represent items that have been placed into the sequence of interest, while the items to the right of the vertical bracket represent items that have not yet been put into the sequence. For example, at level 2 of the search

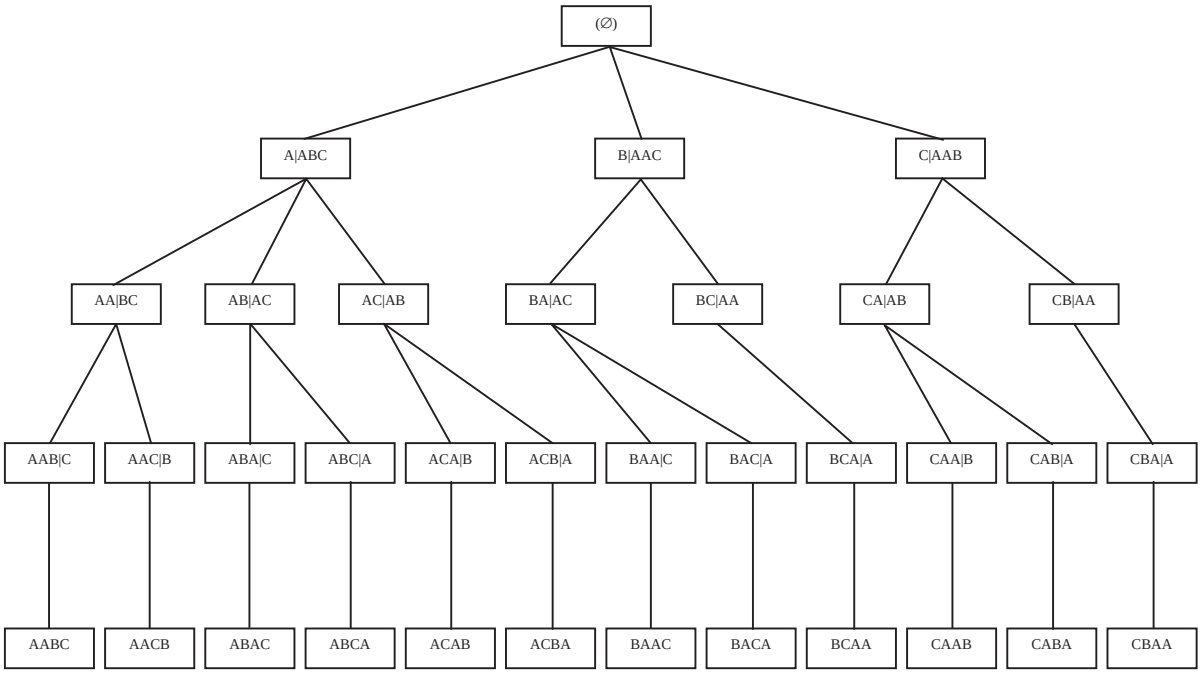

Fig. 2. Full branching solution of example problem.

tree, AC|AB suggests that AC are already in the partial sequence, while (the second) A and B are not—they will be placed in the sequence at levels 3 and 4. Note that level 4 represents all possible sequences, or permutations of this simple problem. It is appropriate to note that beam search is presented here as a means of traversal through the enumeration tree—a way to get to the lowest levels of the tree without having to explore all branches of the tree. One could think of beam search as an efficient means of tree traversal.

Fig. 3 shows a beam search solution to the same example problem where the beam width is $b = 2$ and the depth at which pruning commences is depth $= 2$. At level 2, only the most promising partial sequence (in terms of usage rate) is kept—other partial sequences are pruned. For example, the partial sequences BA|AC and BC|AA are the resultant partial sequences of B|AAC. BA|AC is kept and BC|AA is pruned because BA|AC's usage rate of 1.375 (from Eq. (2)) is less than BC|AA's usage rate of 2.375 (from Eq. (2)). In situations where there is a "tie" for lowest usage rate, the "leftmost node is selected for future branching. This is illustrated by the resultant nodes of A|ABC—AB|AC and AC|AB. Here AB|AC is kept and AC|AB is pruned because AB|AC is left of AC|AB despite the fact that they have the same usage rate.

Another important point to make here is that usage rate, not number of setups, is used to determine which nodes are pruned. The number of setups could be used to determine pruning status, but experimentation showed that using this as a criterion resulted in relatively inferior solutions in terms of efficient frontier performance due to "frontier voids"—this issue is further addressed in the results section of the paper.

Fig. 2 shows the "full" enumeration tree for the example problem, while Fig. 3 shows the "partial" enumeration for the example problem made possible by the beam search approach.

4.2. Example problem

To illustrate the described methodology, an example problem is presented. Consider the situation where five units of item A are required,

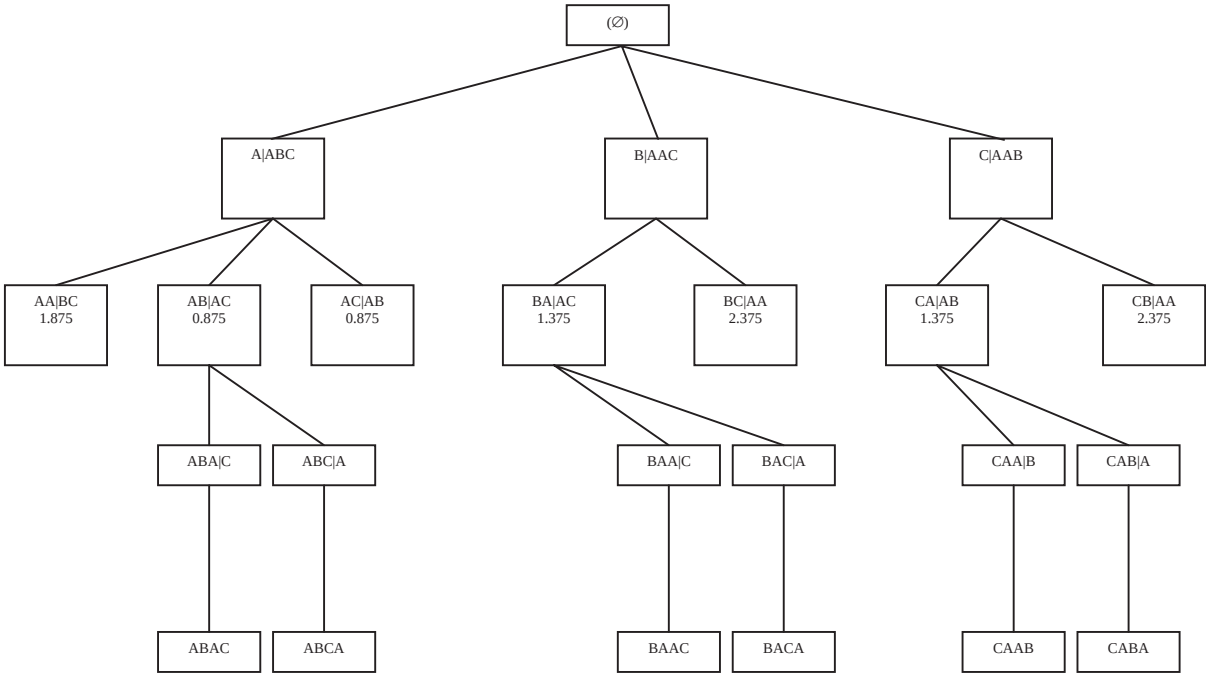

Fig. 3. Beam search solution space to example problem, with $b = 2$ and depth $= 2$.

three units of items B, C, and D are required, and one unit of item E. One possible sequence is AAAAABBBCCCDDDE. All possible sequences for this problem were enumerated—there are 50,450,400 of them. There are as few as five setups possible (the sequence above requires five setups), and as many as 15 possible setups (ABCDABCA-DEABCDA represents an example of this situation). During the enumeration process, the minimum usage rate for each possible number of setups is found. This comprises the optimal efficient frontier. This problem was also addressed via beam search, with a beam width of $b = 2$, and pruning commencing at depth $= 4$. The minimum usage rate for each possible number of setups was also found for this beam search approach. Both efficient frontiers are presented in Fig. 4.

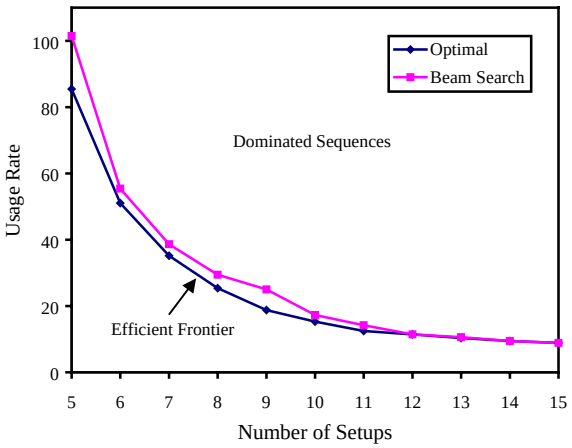

Fig. 4. Efficient frontiers for optimal solution and beam search approach.

As one can see from the figure above, the beam search approach does yield consistently higher usage rates as compared to the optimal solution. But of the 50,450,400 enumerated solutions, only 12,660 of them were found to be superior to those obtained via beam search for each possible number of setups. This places the beam search result in the 99.9749th percentile for this example problem—a near-optimal condition. The beam search solution also took only 1.43 minutes to compute versus the 25.54 minutes required for

complete enumeration—a fraction of the time for complete enumeration.

5. Experimentation and results

5.1. Test problems and parameters

Several test problems were used to evaluate the beam search approach in terms of performance compared to optimality as well as performance in terms of CPU time. Problems were essentially obtained via two different sources. The first source was from a specific wood-processing operation—turning/lathing wooden sticks into vertical members for stairway banisters. There is demand for each type of vertical member, and setup time between differing item is non-negligible. The demand dynamics for these items are consistent with the literature Sumichrast and Russell (1990), the second source—illustrating varying degrees of single item product-mix dominance. These test problems are used to evaluate the methodology presented here. These problems appear in the Appendix. These problems were solved according to varying sets of beam search parameters. A listing of these parameters is provided in Table 1.

5.2. Heuristic performance

In terms of beam search performance, two measures are of interest: objective function performance and CPU time. Objective function performance implies a comparison of the efficient frontier obtained via beam search and the optimal efficient frontier obtained via enumeration. CPU time here means a comparison of required CPU time for the beam search approach and the required CPU time for complete enumeration. These algorithms were run on a system with dual Pentium III processors with a clock speed of 500 Mhz.

A comparison of beam search results and complete enumeration is summarized in Table 2.

The values in Table 2 require some explanation. The CPU ratio is as follows:

$$\text{CPU Ratio} = \frac{\text{Beam Search CPU Time}}{\text{Complete Enumeration CPU Time}}.$$

Lower CPU ratios reflect relatively little beam search CPU time as compared to the CPU time for complete enumeration—lower CPU ratios are desired.

Average inferiority is straightforward—the average amount the usage rate for the beam search solution is in excess of the usage rate for the optimal solution at each level of required setups.

Frontier voids represents the number of times the beam search heuristic was unable to find a sequence for a specific number of setups. Fig. 5 presents a graphic example of a situation where frontier voids are the result of a search.

Here, there are two situations where the search did not find a sequence for a specific number of setups: seven setups and eleven setups. Frontier voids suggest the weakness of the search—the

Table 1
Parameter values used for test problems

Parameter label	Beam width (b)	Pruning depth (depth)
1	1	2
2	2	2
3	2	3
4	2	4
5	2	5
6	3	2
7	3	3
8	2	6

Table 2
Performance measures organized by parameter values

Parameter label	CPU ratio (Std. dev.)	Avg. inferiority % (Std. dev.)	Frontier voids (Std. dev.)
1	0.0623 (0.0908)	19.69 (11.93)	5.667 (2.33)
2	0.2237 (0.3091)	13.77 (5.89)	1.89 (1.02)
3	0.3376 (0.3451)	10.06 (5.32)	0.06 (0.24)
4	0.5030 (0.4630)	7.51 (4.58)	0.00 (0.00)
5	0.7800 (0.6620)	5.49 (3.25)	0.00 (0.00)
6	1.0840 (0.6040)	3.45 (2.87)	0.44 (0.51)
7	1.9760 (1.0930)	2.13 (2.24)	0.06 (0.25)
8	0.9670 (0.6620)	3.21 (2.00)	0.00 (0.00)

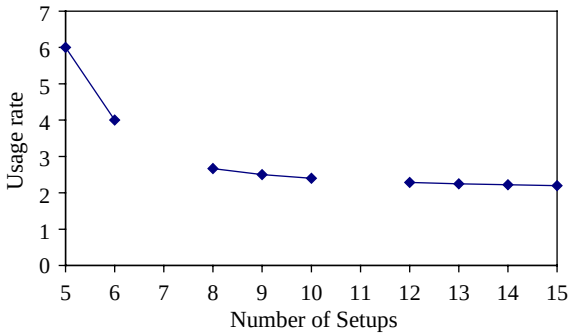

Fig. 5. Example of frontier voids.

inability of the search to find solutions for all possible number of setups. Minimum frontier voids are desired. It should be noted, however, that it is quite possible for sequences obtained via complete enumeration to have frontier voids as well.

From inspection of Table 2, it is clear that the solutions obtained via parameter labels 1 and 2 result in undesirable solutions—there are many frontier voids and the level of inferiority is high. These levels of beam width and pruning depth will be considered no further. One may also notice that parameter labels 6, 7 and 8 result in desirable solutions in terms of average inferiority (near-optimal solutions). The problem with these solutions is that they are CPU intensive—they essentially require as much or more CPU time as the optimal solutions obtained via complete enumeration. As a result, solutions obtained via the beam width and pruning depth of parameter labels 6, 7 and 8 will be considered no further.

As a result of using this "elimination" process, only solutions obtained via the beam-width and pruning depth represented by parameter labels 3, 4 and 5 remain. These solutions have reasonably low levels of inferiority, CPU times are consistently less than unity (which suggests less needed CPU time as compared to complete enumeration), and minimal frontier voids. Solutions obtained by these combinations of b and depth are investigated further—Table 3 shows these results.

Table 3 demonstrates that, in terms of percentile performance, any of these approaches result in near-optimal performance. It does seem, however, that increasing the pruning depth to levels of 4 and/or 5 provides improvements that perhaps do not offset the additional CPU resources. Therefore, for the given problems here, use of a beam width of $b = 2$ and a pruning depth of depth $= 3$ seem most reasonable.

Table 3
Percentiles (and standard deviations) of parameter labels 3, 4 and 5

Parameter label	Demand of 12 items	Demand of 15 items
3 ($b = 2$, depth $= 3$)	99.84% (0.10%)	99.51% (0.95%)
4 ($b = 2$, depth $= 4$)	99.88% (0.11%)	99.54% (0.93%)
5 ($b = 2$, depth $= 5$)	99.89% (0.12%)	99.55% (0.92%)

5.3. Unique and beneficial features of problem structure

This type of problem has some features that can simplify finding desirable solutions for larger problems. The sequences obtained via the presented methodology can be "mapped," or extrapolated to obtain solutions for larger problems. Specifically, replication can be employed to accommodate larger problems. Consider the example presented earlier in the problem definition section of the paper: demand for four units of item A, demand for two units of item B and demand for one unit of item C. Assuming that the following sequence is obtained via the heuristic: AABCAAB, the following sequence could be subsequently obtained via replication if demand for each unique item were tripled: AABCAAB|AABCAAB|AABCAAB.

Furthermore, consider the possibility where a practitioner is interested in integrating an additional unique item into the mix. For example, assume that one unit of unique item D is demanded, given the "tripling" of the original product-mix. Item D could be placed somewhere in the middle of the sequence as follows: AABCAAB|AABCDAAB|AABCAAB. Or, if two units of item D are demanded, then these two units

could be placed somewhere in the sequence, such as the following: AABCAAB|DAABCAAB-D|AABCAAB (if minimal usage rate is desired), or AABCAAB|AABCDDAAB|AABCAAB (if minimal setups are desired).

This replication and extrapolation can be used to retro-fit solutions obtained via the presented methodology to larger problems, which is beneficial considering the combinatorial and memory limitations associated with the presented approach. Wantuck (1989) provides some additional guidelines regarding the "drop-ins" associated with adding new items to the sequence.

6. Conclusion

This paper presented the mixed-model scheduling with setups problem. Two conflicting objectives arise when addressing this problem. These are important objectives with respect to successful JIT implementation. Given these conflicting objectives, an efficient frontier approach has been developed as an aid to decision-makers in evaluating the tradeoffs involved and choosing a final manufacturing sequence. A beam-search heuristic is used to effectively generate the efficient frontiers. Near optimal solutions are obtained with reasonable computational effort. This type of search heuristic, as opposed to others, permits decision-makers to exploit representative components of the entire search space, as opposed to being "at the mercy" of stochastic search mechanisms (i.e., Simulated Annealing, Genetic Algorithms or Tabu Search). As a next step in this research, a formal decision support system might be implemented to allow managers to quickly generate the efficient frontiers for a problem that they might encounter.

This research has its limitations. For beam search, it is necessary to hold "in memory" all nodes of the current level so that relevant information can be passed from the parent node to the child nodes. While the heuristic performs well for the problem sets investigated in this paper, as problem size increases it may be constrained by the amount of computer memory available for processing. The authors are currently addressing this concern as an extension of this study. Another opportunity for future research would be to implement some Artificial Neural Network approaches to this type of problem. Specifically, Kohonen's self-organizing maps Kohonen (1990) and Hopfield and Tank (1985) approaches could be adapted to address this dual-objective decision-making problem.

Appendix

See Tables 4 and 5.

Table 4
Problem set 1: (number of each product type in product mix—total demand is 12)

Problem	Item 1	Item 2	Item 3	Item 4	Item 5	Total sequences
B	8	1	1	1	1	11,880
C	7	2	1	1	1	47,520
D	6	3	1	1	1	110,880
E	6	2	2	1	1	166,320
F	5	3	2	1	1	332,640
G	5	2	2	2	1	498,960
H	4	3	2	2	1	831,600
I	4	4	2	1	1	415,800
J	3	3	2	2	2	1,663,200

Table 5
Problem set 2: (number of each product type in product mix—total demand is 15)

Problem	Item 1	Item 2	Item 3	Item 4	Item 5	Total sequences
B	11	1	1	1	1	32,760
C	10	2	1	1	1	180,180
D	9	3	1	1	1	600,600
E	7	5	1	1	1	2,162,160
F	7	3	2	2	1	10,810,800
G	6	3	3	2	1	25,225,200
H	5	3	3	3	1	50,450,400
I	4	3	3	3	2	126,126,000
J	3	3	3	3	3	168,168,000

References

Bard, J., Dar-El, E., Shtub, A., 1992. An analytic framework for sequencing mixed-model assembly lines. International Journal of Production Research 30, 35–48.

Bolat, A., 1994. Sequencing jobs on an automobile assembly line: Objectives and procedures. International Journal of Production Research 32, 1219–1236.

De, P., Ghosh, J.B., Wells, C.E., 1992. Heuristic estimation of the efficient frontier for a bi-criteria scheduling problem. Decision Sciences 23, 596–609.

Fox, M.J., 1983. Constraint-directed search: A case study of job-shop scheduling. Unpublished Doctoral Dissertation, Carnegie-Mellon University, Pittsburgh, 1983.

Ghosh, S., Gagnon, R., 1989. A comprehensive literature review and analysis of the design, balancing and scheduling of assembly systems. International Journal of Production Research 37, 620–637.

Gilbert, J.P., 1990. The state of JIT implementation and development in the USA. International Journal of Production Research 28, 1099–1109.

Hopfield, J.J., Tank, D.W., 1985. "Neural" computation of decisions on optimization problems. Biological Cybernetics 52, 141–152.

Huson, M., Nanda, D., 1995. The impact of just-in-time manufacturing on firm performance in the US. Journal of Operations Management 12, 297–310.

Inman, R., Bulfin, R., 1991. Sequencing JIT mixed-model assembly lines. Management Science 37, 901–904.

Kohonen, T., 1990. The self-organizing map. Proceedings of the IEEE 78, 1464–1480.

McMullen, P.R., Frazier, G.V., 2000. A simulated annealing approach to mixed-model sequencing with multiple objectives on a JIT line. IIE Transactions 32, 679–686.

Miltenburg, J., Sinnamon, G., 1989. Scheduling mixed-model multi-level just-in-time production systems. International Journal of Production Research 27, 1487–1509.

Miltenburg, J., Sinnamon, G., 1992. Algorithms for scheduling multi-level just-in-time production systems. IIE Transactions 24, 121–130.

Miltenburg, J., Sinnamon, G., 1995. Revisiting the mixed-model multi-level just-in-time scheduling problem. International Journal of Production Research 33, 2049–2052.

Miltenburg, J., 1989. Level schedules for mixed-model assembly lines in just-in-time Production systems. Management Science 35, 192–207.

Monden, Y., 1983. Toyota Production System, The Institute of Industrial Engineers, Norcross, GA.

Nair, S.K., Thakur, L.S., Wen, K., 1995. Near optimal solutions for product line design and selection: Beam search heuristics. Management Science 41, 767–785.

Norvig, P., 1992. Paradigms of Artificial Intelligence Programming Case Studies in Common LISP. Morgan-Kaufman Publishers, San Francisco, CA.

Ow, P.S., Morton, T.E., 1988. Filtered beam search in scheduling. International Journal of Production Research 26, 35–62.

Ow, P.S., Morton, T.E., 1989. The single machining early/tardy problem. Management Science 35, 177–191.

Ow, P.S., Smith, S.F., 1988. Viewing scheduling as an opportunistic problem-solving process. Annals of Operations Research 12, 85–108.

Sumichrast, R.T., Russell, R.S., 1990. Evaluating mixed-model assembly line sequencing heuristics for just-in-time production systems. Journal of Operations Management 9, 371–390.

Sumichrast, R.T., Russell, R.S., Taylor, B.W., 1992. A comparative analysis of sequencing procedures for mixed-model assembly lines in a just-in-time production system. International Journal of Production Research 30, 199–214.

Tamura, T., Long, H., Ohno, K., 1999. A sequencing problem to level part usage rates and work loads for a mixed-model assembly line with a bypass subline. International Journal of Production Research 61, 557–564.

Wantuck, K.A., 1989. Just in Time for America. KWA Media, Smithfield, MI.

White, R.E., Ruch, W.A., 1990. The composition and scope of JIT. Operations Management Review 7, 9–18.

White, R.E., 1993. An empirical assessment of JIT in US manufacturers. Production and Inventory Management Journal 34, 38–42.

Xiaobo, K., Ohno, A., 1994. A sequencing problem for a mixed-model assembly line in a JIT production system. Computers and Industrial Engineering 27, 71–74.

Xiaobo, Z., Zhou, Z., Asres, A., 1999. A note on Toyota's goal of sequencing mixed models on an assembly line. Computers & Industrial Engineering 36, 57–65.

Yano, C., Bolat, A., 1989. Survey, development and applications of algorithms for sequencing paced assembly lines. Journal of Manufacturing and Operations Management 2, 172–198.