Resequencing and Feature Assignment on an Automated Assembly Line

Maher Lahmar, Hakan Ergan, and Saif Benjaafar, *Member, IEEE*

***Abstract*—We consider the problem of resequencing a prearranged set of jobs on a moving assembly line with the objective of minimizing changeover costs. A changeover cost is incurred whenever two consecutive jobs do not share the same feature. Features are assigned from a set of job-specific feasible features. Resequencing is limited by the availability of offline buffers. The problem is motivated by a vehicle resequencing and painting problem at a major U.S. automotive manufacturer. We develop a model for solving the joint resequencing and feature assignment problem and an efficient solution procedure for simultaneously determining optimal feature assignments and vehicle sequences. We show that our solution approach is amenable to implementation in environments where a solution must be obtained within tight time constraints. We also show that the effect of offline buffers is of the diminishing kind with most of the benefits achieved with very few buffers. This means that limited resequencing flexibility is generally sufficient. Furthermore, we show that the value of resequencing is sensitive to the feature density matrix, with resequencing having a significant impact on cost only when density is in the middle range.**

***Index Terms*—Assembly line design, feature assignment, sequencing with setups.**

I. Introduction

WE CONSIDER the problem of resequencing a prearranged set of jobs on a moving assembly line with the objective of minimizing changeover costs. A changeover cost is incurred whenever two consecutive jobs do not share the same feature. Features are assigned from a job-specific set of feasible features. Resequencing is limited by the availability of offline buffers where a job removed from the line could be temporarily stored before it is reinserted. Because jobs are continuously moving, resequencing is limited to jobs that are upstream from the offline buffer. Since jobs may visit multiple departments before leaving the system, a sequence that is optimal for one department is usually suboptimal for most other departments. Consequently, there is a need to resequence jobs for each department.

Problems where jobs must be resequenced using offline buffers are frequently encountered in industries where a continuous material transfer system is used, and multiple products, or products with varying features, are produced on the same line. In our case, the problem is motivated by a vehicle-sequencing problem at a North American Ford truck assembly plant. In the plant, vehicles must be resequenced upon leaving the body shop and before entering the paint area. In the paint area, vehicles undergo a series of coating, painting, and drying operations. If two consecutive vehicles are painted different colors, a significant changeover cost is incurred since the current paint must be flushed out and disposed of, and paint nozzles must be thoroughly washed and cleaned with solvents. The changeover cost is primarily determined by the volume of paint that must be purged from an intermediate pipe that connects the paint nozzle to the paint tanks. Although the volume of paint that is purged is always the same (approximately one gallon), the cost of different paints can vary widely. Color costs range from $27/gallon (basic black) to $122/gallon (bright amber). This makes it desirable to form large blocks of the expensive colors while possibly incurring the cost of more frequent color changeovers of the cheaper ones. Note that the cost of changing colors is not sequence dependent (e.g., the cost of going from darker to lighter colors is not more expensive than going from lighter to darker colors). New paint technology has recently removed this dependency [1].

Flexibility in assigning colors to vehicles is limited by current demand requirement and available vehicle body type. Therefore, the set of feasible colors (features) for two consecutive vehicles can vary significantly. Because of space constraints and strict requirements on assembly line speed, there is currently little opportunity to minimize paint changeover costs by altering the sequence of vehicles. Therefore, changeover costs are being effected solely through color assignment. However, the plant is contemplating using a limited number of offline buffers, or pull-off tables, to allow for improved vehicle sequencing. Because pull-off tables are expensive to implement and operate, only a few can be introduced.

As shown in Fig. 1, a pull-off table will allow a vehicle to be removed from the line with little interruption to line flow and reinserted later. The line is designed to allow pulling and reinsertion at any point in the sequence without significantly affecting line speed (pull-off tables were originally designed to pull and reinsert vehicles that require rework) [1]. However, because pull-off tables are stationary, a pulled vehicle can only be inserted upstream from its initial position. Also, because a pull-off table serves as a temporary storage buffer for pulled vehicles, a pull-off table is unavailable to remove other vehicles until the current one has been reinserted.

Although the plant produces approximately 1100 vehicles per day, only a small block of vehicles (15 or so in our applica-

Manuscript received November 12, 2001; revised April 28, 2002. This paper was recommended for publication by Associate Editor C. Chu and Editor N. Viswanadham upon evaluation of the reviewers' comments. This work was supported in part by the National Science Foundation under Grant DMII-9908437 and in part by the Ford Motor Company.

M. Lahmar and S. Benjaafar are with the Graduate Program in Industrial Engineering, Department of Mechanical Engineering, University of Minnesota, Minneapolis, MN 55455 USA (e-mail: maher@ie.umn.edu; saif@ie.umn.edu).

H. Ergan is with USAirways, Pittsburgh, PA 15275 USA (e-mail: hakan_ergan@usairways.com).

Digital Object Identifier 10.1109/TRA.2002.807556

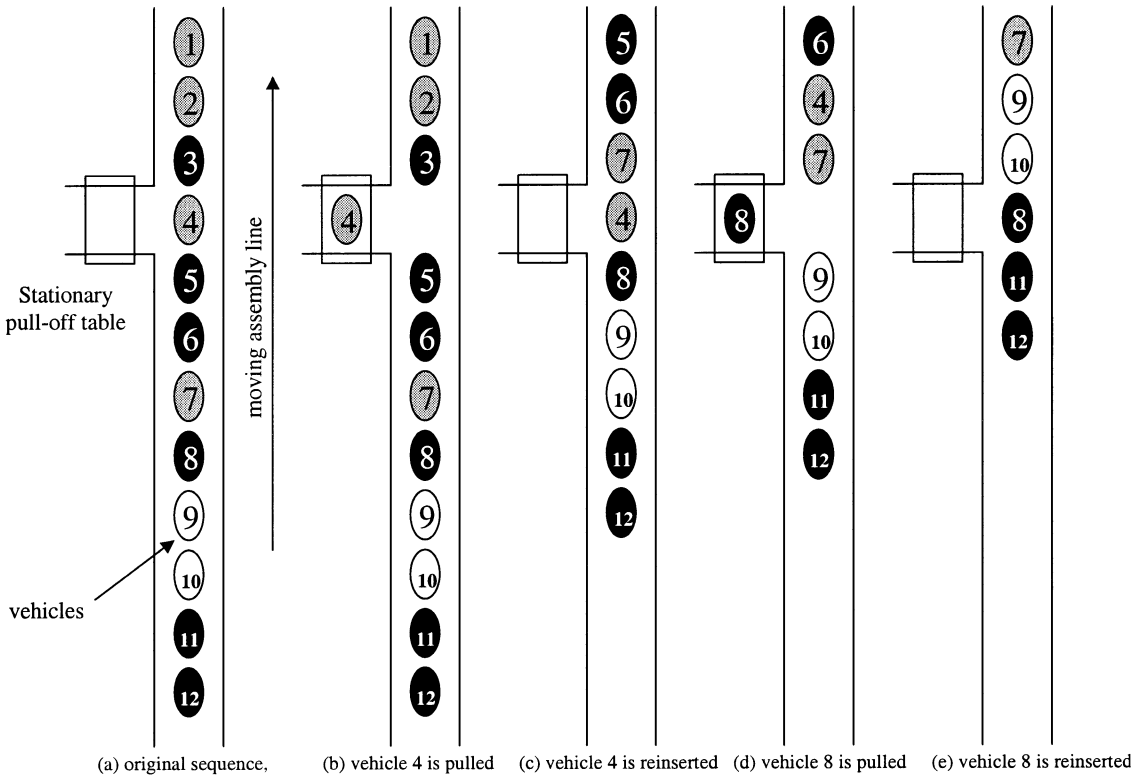


Fig. 1. Vehicle resequencing using pull-off tables.

tion) are considered for resequencing at a time. This is due to the disruptions of the original sequence that take place at the body shop, making it difficult to predict too far in advance the input sequence to the paint area. These disruptions have various causes including rework, breakdowns, and multiplicity of routings within the body shop. The resulting limited look ahead means that resequencing and color assignment have to be carried out successively for one block of vehicles at a time.

The color-assignment matrix for the current block usually reflects the current product mix (i.e., demand for combinations of body style and color). For each vehicle, the matrix provides the set of colors for which there is currently a demand. Since any feasible color could theoretically be selected during the painting process, this may create imbalances in the product mix. However, these imbalances are short lived, since they are taken into account when updating the color-assignment matrix for subsequent blocks. In other words, an overproduction of a particular "body style/color" combination will eliminate this combination from consideration in future periods. Since the plant produces over 1100 vehicles per day and each body style/color combination has a demand significantly greater than 15 vehicles, the product mix will always be balanced at the end of the day.

The introduction of resequencing buffers to a moving assembly line poses a number of challenges. Decisions must be made regarding which vehicles must be pulled and where they should be reinserted. Simultaneously, decisions must be made regarding which color should be assigned to each vehicle given the limited assignment flexibility available to each vehicle. At a strategic level, a decision must be made regarding how many pull-off tables are desirable or affordable and how the value of these tables is affected by various system parameters. In this paper, we address several of these challenges. In particular, we present an integrated model for the joint problem of color assignment and vehicle resequencing, where the objective is to minimize color changeover costs. Because resequencing and color assignment must be carried out for successive blocks of vehicles, we present an efficient solution approach that is amenable to real-time implementation. We also examine the impact of multiple offline buffers on performance and study the relationship between resequencing ability and system parameters, such as block size and density of the color-assignment matrix. Using numerical examples, we show that the effect of offline buffers is of the diminishing kind with most of the benefits of total resequencing flexibility achieved with only a few buffers. For the case of Ford, we show that significant savings, in excess of $1.2 million per paint line per year, can be realized with the introduction of only a single buffer. We also show that the value of resequencing is dependent on the density of the color-assignment matrix, with resequencing having the greatest impact when density is in the middle range.

The organization of the paper is as follows. In Section II, we present a brief review of relevant literature. In Section III, we present a formulation of the joint resequencing and color assignment problem. In Section IV, we provide a polynomial-time dynamic programming (DP) algorithm for solving problems with a single offline buffer. We also present a decomposition approach

for solving problems with multiple buffers. We describe several structural properties of the problem that allow us to significantly enhance the computational efficiency of our algorithm. In Section V, we present numerical results using both actual and randomly generated data. We examine the impact of various parameters on the quality of the solution, such as number of buffers (pull-off tables), number of vehicles and colors, and varying densities of the color-assignment matrix. We also study the impact of multiple buffers on cost and show that the benefits of multiple buffers are of the diminishing kind with most of the cost reduction realized with one buffer.

II. Literature Review

Despite the fact that there is a large body of literature on the sequencing of paced assembly lines (see [2] and [3] for a general review), most of it assumes that the original sequence of jobs remains fixed. Therefore, the primary concern is determining a single job sequence for the entire line for an available (or recurring) set of jobs, where the objective is, typically, to balance workload among the different processing departments [4], [5]. Few papers, especially those that deal with mixed assembly lines, consider sequence-dependent setups. For example, Burns and Daganzo [6] and Bolat *et al.* [7] consider lines where different jobs have different features or options and a setup is incurred whenever two jobs with different options follow each other. They develop heuristics for sequencing these jobs with the objective of minimizing total changeover costs. However, they do not consider the issue of resequencing, and assume that jobs have unique features. The issue of sequencing jobs with options is also discussed in [5], which considers cases where jobs with different options have different processing times. However, they assume there is no setup between jobs with different options. Myron [1] examines the effect of forming large blocks of same-color vehicles on paint changeover cost at an automotive assembly plant. Using discrete event simulation, he shows that a simple block protection rule, when coupled with pre- and post-sequencing using a fully flexible automated storage and retrieval system (AS/RS), could significantly reduce these costs.

The issue of sequence-dependent setups has been addressed extensively in the traditional scheduling literature. Most of this literature considers a single machine problem with multiple jobs, where individual jobs may belong to different families. A setup time is incurred whenever two consecutive jobs belong to different families. The individual jobs, irrespective of family membership, may carry different weights and have different due dates. The objective is to determine a sequence of jobs that optimizes one or more performance measures—typically, a function of job completion time (e.g., maximum lateness, weighted completion time, or weighted tardiness). Examples of this work include [8]–[11]. In general, scheduling with sequence-dependent setups is NP-hard, with polynomial algorithms available only for a few special cases [12], [13].

In all of the above literature, it is assumed that there is full flexibility in how jobs are sequenced. It is also assumed that setups are family or lot specific, with family or lot membership being known. The problem we treat in this paper is different from this literature in two respects. First, we assume that there is only limited flexibility in how jobs can be resequenced. Second, we allow some flexibility in assigning attributes that determine family membership among jobs. To our knowledge, this is the first paper to consider the problem of joint attribute assignment and resequencing when there is limited resequencing flexibility and job-dependent setups.

III. Problem Formulation

For ease of discussion, and without loss of generality, we will refer to jobs as vehicles, attributes as paint colors, and offline buffers as pull-off tables. Before we present our formulation of the joint resequencing and color assignment problem, we introduce the following notation.

m: number of vehicles
n: number of colors
N: number of pull-off tables

$$a_{i,k} = \begin{cases} 1, & \text{if vehicle } i \text{ can be assigned color } k \\ 0, & \text{otherwise} \end{cases}$$

c_k: changeover cost from color k

$$x_{i,k,h} = \begin{cases} 1, & \text{if vehicle } i \text{ is assigned color } k \text{ and its final position is } h \\ 0, & \text{otherwise} \end{cases}$$

and

$$\delta_{k,h} = \begin{cases} 1, & \text{if color } k \text{ is changed after painting the vehicle at position } h \\ 0, & \text{otherwise.} \end{cases}$$

The resequencing and color-assignment problem for a block of n vehicles with N pull-off tables can be formulated as follows:

$$\text{Minimize} \quad z = \sum_{h=1}^{m}\sum_{k=1}^{n} c_k \delta_{k,h} \tag{1}$$

Subject to

$$\delta_{k,h} \geq \sum_{i=1}^{m} x_{i,k,h} - \sum_{i=1}^{m} x_{i,k,h+1}, \quad k = 1, \ldots, n;\ h = 1, \ldots, m-1 \tag{2}$$

$$\delta_{k,m} \geq \sum_{i=1}^{m} x_{i,k,m}, \qquad k = 1, \ldots, n \tag{3}$$

$$\sum_{i=1}^{m}\sum_{k=1}^{n} x_{i,k,h} = 1, \qquad h = 1, \ldots, m \tag{4}$$

$$\sum_{h=1}^{m}\sum_{k=1}^{n} x_{i,k,h} = 1, \qquad i = 1, \ldots, m \tag{5}$$

$$x_{i,k,h} \leq a_{i,k} \quad i = 1, \ldots, m;\ h = 1, \ldots, m;\ k = 1, \ldots, n \tag{6}$$

$$x_{i,k,h} = 0 \quad \forall\, i > h + N; \quad h = 1, \ldots, m;\ k = 1, \ldots, n \tag{7}$$

$$x_{i,k,h} \in \{0, 1\}, \quad i = 1, \ldots, m;\ h = 1, \ldots, m;\ k = 1, \ldots, n \tag{8}$$

$$\delta_{k,h} \in \{0, 1\}, \qquad h = 1, \ldots, m;\ k = 1, \ldots, n. \tag{9}$$

The objective function minimizes the cost of color changeovers. Constraints (2) ensure that a changeover

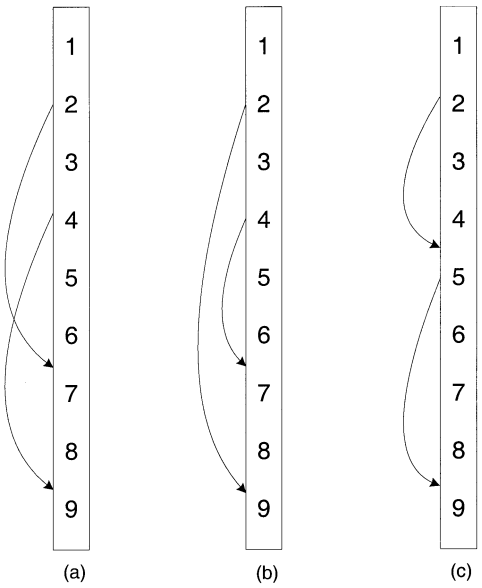


Fig. 2. Example pulling scenarios.

cost is incurred only if two consecutive vehicles are assigned two different colors. Constraints (3) guarantee that paint is flushed after the last vehicle is painted. Constraints (4) ensure that only one vehicle is assigned to each position in the final sequence. Constraints (5) require that each vehicle be assigned only one position and only one color. Constraints (6) restrict the color assignment to the set of feasible color options.

Constraints (7) guarantee that, at most, N vehicles are pulled at a time and that the pulling and reinsertion occur in the proper order. This is accomplished by first noting that if a vehicle is pulled and later reinserted, its position in the final sequence will be lower or remain the same ($h \geq i$ in the index of the variable $x_{i,k,h}$). On the other hand, vehicles that are not pulled will retain the same position or will advance in the sequence ($h \leq i$). The number of positions a vehicle could advance depends on the number of pull-off tables. For a system with N pull-off tables, a vehicle could advance by, at most, N positions. In contrast, a pulled vehicle could be placed anywhere downstream from its original sequence.

Note that vehicles that are pulled by the same pull-off table are always reinserted in the same order they were pulled. Doing otherwise would result in more than one vehicle being pulled by the same table at the same time which, in turn, would result in some vehicles advancing by more than N positions in violation of (7). These issues are illustrated in Fig. 2 for a system with one pull-off table, where the series of actions in scenarios (a) and (b) are always unfeasible (i.e., constraints (7) are not satisfied) and only scenarios of type (c) are possible. In scenario (a), vehicle 2 is pulled and inserted after vehicle 6, and vehicle 4 is pulled and inserted after vehicle 8. This scenario is unfeasible since it results in vehicles 5 and 6 advancing by two positions in the final sequence (they would be, respectively, in positions 3 and 4). In scenario (b), vehicle 2 is pulled and inserted after vehicle 8, and vehicle 4 is pulled and inserted after vehicle 6. This scenario is unfeasible since it results again in vehicles 5 and 6 advancing by two positions. In scenario (c), vehicle 2 is pulled and inserted after vehicle 4, and vehicle 5 is pulled and inserted after vehicle 8, which is feasible. Solutions that satisfy (7) are always feasible. An unfeasible solution would require the usage of more than N pull-off tables at time. However, this cannot occur since, by virtue of (7), the most positions a vehicle can advance is N.

Although the values of the variables $x_{i,k,h}$ reflect the assigned color and final position for each vehicle, there could be a need during implementation to explicitly identify which vehicles are pulled. This can be easily retrieved from the solution to the model in (1)–(9) as follows. Let

$$y_{i,h} = \begin{cases} 1, & \text{if vehicle } i \text{ is pulled and reinserted in position } h \\ 0, & \text{otherwise.} \end{cases}$$

Then,

$$y_{i,h} = \begin{cases} 1, & \text{if } \sum_{i'>i} \sum_{k} \sum_{k'} \sum_{h'<h} x_{i',k',h'} x_{i,k,h} > 0 \\ 0, & \text{otherwise.} \end{cases} \quad \forall\, i \neq m, \forall\, h \neq 1 \tag{10}$$

$$y_{i,1} = 0 \qquad i = 1, \ldots, m \tag{11}$$

$$y_{m,h} = 0 \qquad h = 1, \ldots, m. \tag{12}$$

The above follows from noting that a vehicle is pulled if and only if at least one vehicle that was behind it in the original sequence is ahead of it in the final sequence. Note that with more than one pull-off table, it is not always true that if a vehicle i is pulled, then it is placed in a final position immediately behind a vehicle i that was originally behind i in the initial sequence. Expressions (11) and (12) treat two exceptional cases. The first is when the position of a vehicle i is first in the final sequence. In that case, vehicle i could not have been pulled (a vehicle could not move up in the sequence through a pulling action). The second case is when a vehicle is in the last position (position m) in the original sequence. In that case, since the vehicle is already last, it cannot be pulled and reinserted behind other vehicles.

In general, the resequencing and color (feature) assignment problem is NP-hard. For example, in the case when there is a sufficient number ($N \geq m-1$) of pull-off tables to allow for full resequencing flexibility, the problem reduces to a set covering problem (see Appendix I) which is known to be NP-hard [14], [15]. This means that solving even a relatively small problem can be computationally demanding. Computational effort in our case is particularly a concern since decisions must be made under tight time constraints. The fact that vehicles are continuously moving, with approximately one-minute spacing, means that solutions must be obtained in few seconds.

Fortunately, as we show in the next section, the problem with a single pull-off table can be solved in polynomial time. Therefore, we propose a decomposition approach where we successively solve a series of problems with a single pull-off table. We benchmark the solution we obtain using our decomposition approach against optimal solutions for solvable problems. We also develop upper and lower bounds on the solution and show that our approach yields reasonably good solutions. We show that in our application, the decomposition has a limited effect on the quality of our solution, since pull-off tables are expensive to implement and, at most, one or two can be accommodated on a single line. More importantly, we show that there is, in general,

a diminishing effect to having multiple tables, and that most of the benefits of resequencing occurs with a single pull-off table.

Solving the problem with N pull-off tables using this decomposition approach can be clearly done in polynomial time. The complexity of the algorithm is of order $O(Nm^2n^4)$. More significantly, our numerical results show that a solution for problem sizes of interest can be solved in few seconds. In a recent paper, Lahmar and Benjaafar [16] develop an exact dynamic programming algorithm to solve the resequencing problem without color assignment (i.e., $n = 1$). They show that the maximum complexity of their algorithm is of order $O(m^N)$. That is, computational effort increases exponentially in the number of pull-off tables but polynomially in m.

IV. Solution Algorithm

We decompose the problem with N pull-off tables into a series of N problems with a single pull-off table. The problems are solved in an iterative fashion where the final sequence obtained in iteration i serves as the initial sequence for iteration $i+1$. At each iteration i, we solve optimally and simultaneously the resequencing and color-assignment problem with the ith pull-off table. Although the decomposition approach does not guarantee optimality, it is computationally efficient and does lead, as we show in Section V, to high-quality solutions.

Before we consider a general solution to the resequencing and color-assignment problem with a single pull-off table, we develop an optimal solution for the color-assignment problem with a fixed sequence. We will use the solution to this subproblem in constructing an integrated solution to the original resequencing and color-assignment problem.

A. Color-Assignment Problem

Given a fixed sequence of vehicles, the color-assignment problem, as we show in *Proposition 1*, can be formulated as a shortest-path problem and solved in polynomial time.

Proposition 1: The color-assignment problem can be formulated as a shortest-path problem in a staged directed network. A solution can be obtained using an algorithm of maximum complexity of order $O(mn^2)$.

Proof: The color-assignment problem can be represented by a staged directed network, where each stage represents a vehicle. Nodes at each stage represent the set of feasible colors available to the vehicle at that stage. The cost of moving from one node to another is the changeover cost from one color to another. Clearly, an optimal solution is given by the shortest path through the network.

An efficient solution to the color-assignment problem can be found using a DP algorithm with the following recursive function:

$$f_i^*(x_i) = \min_{x_{i-1}} \{c_{x_{i-1},\,x_i} + f_{i-1}^*(x_{i-1})\} \tag{13}$$

where x_i refers to a specific node (color) in stage i, $c_{x_{i-1},\,x_i}$ is the changeover cost from node $i-1$ to node i, and $f_i^{(x_i)}$ is the cost of the optimal sequence from the starting node to node i. Since at each stage, a maximum of n^2 solutions are evaluated, the algorithm has a maximum complexity of order $O(mn^2)$.

B. Resequencing and Color-Assignment Problem

Similar to the color-assignment problem, we can show that the resequencing and color-assignment problem with a single pull-off table can be formulated as a shortest-path problem and solved in polynomial time.

Proposition 2: The resequencing and color-assignment problem can be formulated as a shortest-path problem in a multistage directed network. A solution can be obtained using an algorithm of maximum complexity $O(m^2n^4)$.

Proof: In order to take advantage of the structure of the problem and to allow for efficient solution, we define our network as follows. As in the color-assignment problem, each stage corresponds to a vehicle in the original sequence. Within each stage, we define two types of nodes. The first type corresponds to the decision of not pulling a vehicle and assigning it a specific color. We use the notation $x_k(n,\, i)$ to refer to a node representing a vehicle in stage k which is not pulled and is assigned color i. The second type of nodes corresponds to the decision of pulling a vehicle and assigning it color i. Since we know that if vehicle k (in the initial sequence) is pulled, then vehicle $k+1$ (in the initial sequence) cannot be pulled, we further specify these nodes s by differentiating them according to the color assigned to vehicle $k+1$. We refer to a node representing a vehicle in stage k that is pulled and assigned color i, given that vehicle $k+1$ is assigned color j as $x_k(p, i,\, j)$. This representation, as we show in Section IV-C, allows us to take advantage of several structural properties of the problem which significantly reduce the computational effort required in obtaining an optimal solution. Nodes for an example system are shown in Fig. 3.

Arcs in the network are defined only between nodes that represent vehicles that can be in neighboring locations in the final sequence. Specifically, from nodes $x_k(n,\, i)$, the feasible arcs are either to nodes $x_{k+1}(n,\, i')$ or $x_{k+1}(p,\, i',\, j')$. In the first case, the arc cost, $c_{x_k(n,\,i),\,x_{k+1}(n,\,i')} = c_{ii'}$, is the changeover cost from color i to color i'. In the second case, $c_{x_k(n,\,i),\,x_{k+1}(p,\,i',\,j')} = c_{ij'}$ is the changeover cost from color i to color j'. Since in this second case, vehicle $k+1$ is pulled, vehicle k and $k+2$ are in neighboring locations. The maximum number of arcs that could originate from each $x_k(n,\, i)$ node at the first $m-2$ stages is $n+n^2$. Since no vehicles can be pulled at the terminal stage m, a maximum of n arcs could originate from each node $x_{m-1}(n,\, i)$ in stage $m-1$. The maximum total number of these arcs in the network is, therefore, $(m-2)n(n+n^2)+n^2$. The actual number of arcs is usually significantly smaller, since not all colors are feasible for every vehicle.

From nodes $x_k(p,\, i,\, j)$, there are no feasible arcs to any nodes in stage $k+1$, since if vehicle k is pulled it cannot be inserted before a vehicle in stage $k+1$. However, there are feasible arcs from nodes $x_k(p,\, i,\, j)$ to all the nodes in subsequent stages. Stages $m-2$, $m-1$, and m represent special cases, since vehicles at stage $m-1$, if pulled, become last in the sequence, while vehicles in stage m cannot be pulled. It is not too difficult to show that the maximum number of such arcs in the network is given by $n^3(m-1)(m-2)/2+n^4(m-2)(m-3)/2$ (see Appendix II for proof).

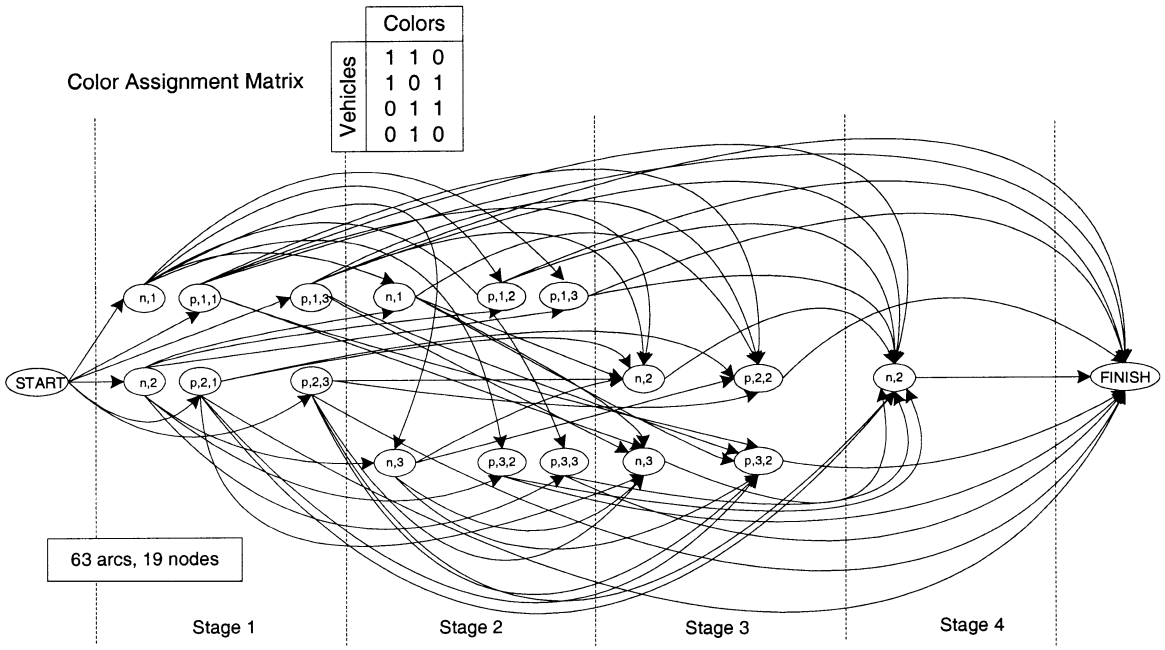


Fig. 3. Example network representation.

An arc from node $x_k(p, i, j)$ to node $x_l(n, i')$ would indicate that vehicle k is inserted before vehicle l, which has already been assigned color i'. Therefore, the cost of an arc from node $x_k(p, i, j)$ to node $x_l(n, i')$ is the cost of changing over from color i to color i' plus the cost of optimally assigning colors to the block of vehicles formed by vehicles $k+1$ (in the original sequence) and the pulled vehicle that has been inserted before vehicle l, given that the first vehicle in this block is assigned color j and the last one is assigned color i. The optimal assignment cost for the block of vehicles is obtained by solving a color-assignment problem using the DP algorithm discussed in Section IV-A. Note that the changeover cost from the last vehicle in the block is incurred only if color i is different from color i'.

An arc from node $x_k(p, i, j)$ to node $x_l(p, i', j')$ indicates that vehicle k is inserted before vehicle l, but vehicle l is pulled. Therefore, vehicle k ends up being behind vehicle $l+1$. The arc cost from node $x_k(p, i, j)$ to node $x_l(p, i', j')$ is calculated, similar to the previous case, by determining the cost of the optimal color assignment of a block of vehicles formed by vehicle $k+1$ and vehicle $l+1$. Again, the changeover cost from color i is incurred only if color i is different from color j'. Finally, an arc from any node, either $x_k(n,i)$ or $x_k(p, i, j)$, to the finishing node indicates that vehicle k is last in the final sequence. The cost of this arc is that of color i. Once the set of nodes and arc costs have been specified, an optimal solution can be efficiently obtained using a DP algorithm with the following recursive function:

$$f_l^*(x_l) = \min_{x_k \in P_l} \{c_{x_k, x_l} + f_k^*(x_k)\} \tag{14}$$

where x_k refers to a specific node in stage k, c_{x_k, x_l} is the changeover cost from node x_k to node x_l, $f_k^*(x_k)$ is the cost of the optimal sequence from the starting node to node x_k, and P_l is the set of nodes with feasible arcs from nodes x_k to x_l. Note that in our case, the set P_l always consists only of nodes from stages $l-1, l-2, \ldots 1$. This is important because it guarantees that the optimal cost $f_k^*(x_k)$ for each node $x_k \in P_l$ has been previously calculated.

Given that an upper bound on the total number of arcs is given by

$$\eta = (m-2)n(n+n^2) + n^2 + n^3(m-1)(m-2)/2 + n^4(m-2)(m-3)/2 \tag{15}$$

which implies a complexity of order $O(m^2n^4)$, our DP has a maximum complexity that is polynomial in m and n. Therefore, a solution can always be obtained in polynomial time. As we discuss next, the computational efficiency can be further enhanced by taking advantage of a number of structural properties of the problem.

C. Computational Enhancements

The efficiency of the DP can be further improved by taking advantage of the following structural properties of the problem.

Property 1: If vehicle k is not pulled and is assigned color i and vehicles $k+1, k+2, \ldots, k+b$ are also not pulled and can be assigned color i, then assigning color i to vehicles $k+1, \ldots, k+b$ is optimal.

Proof: The proof follows from noting that since vehicle k must be painted with color i, a changeover from paint i would have to be eventually incurred, regardless of how vehicles $k+1, \ldots, k+b$ are painted. Since vehicles $k+1, \ldots, k+b$ cannot be resequenced, painting them with color i contributes to the final cost, at most, as much as any other solution. *Property 1* is also applicable in reverse, as described in *Property 2*.

Property 2: If k is not pulled and is assigned color i and vehicles $k-1, \ldots, k-b$ are also not pulled and can be assigned

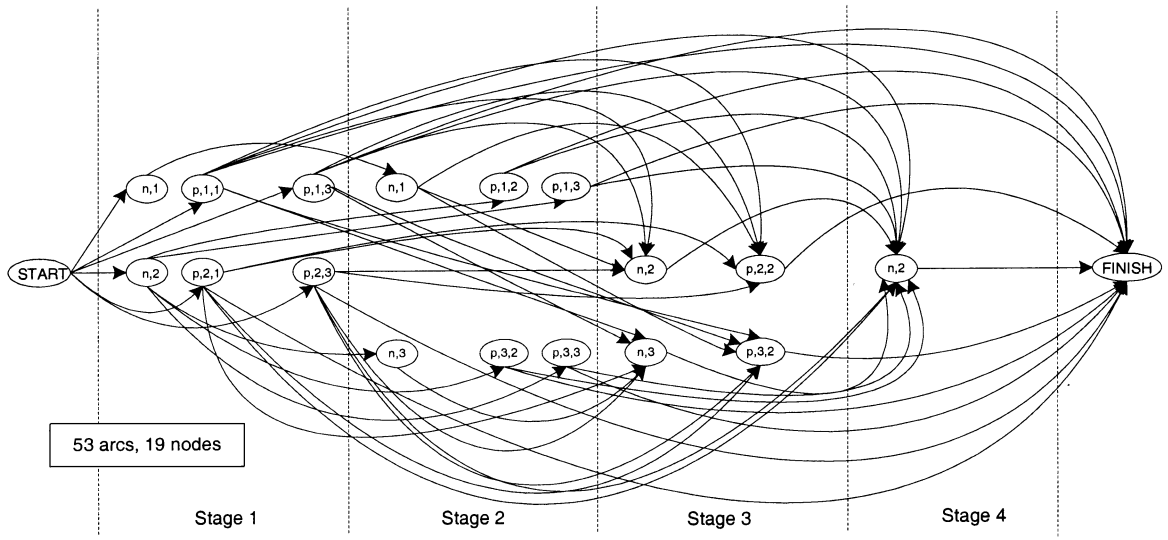


Fig. 4. Effect of applying *Properties 1* and *2* on problem complexity.

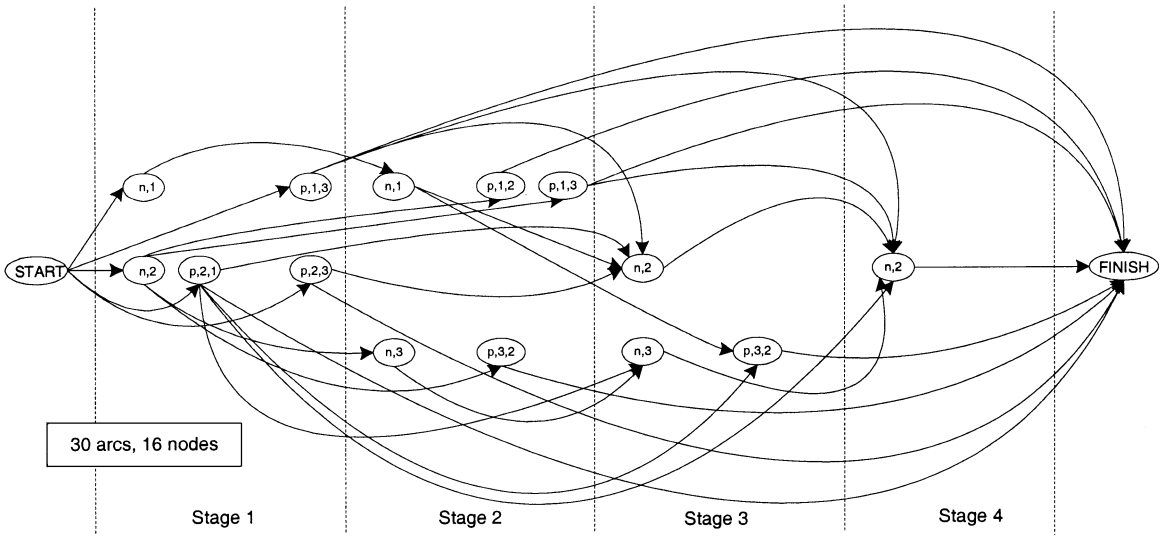


Fig. 5. Effect of applying *Properties 3* and *4* on problem complexity.

color i, then assigning color i to vehicles $k-1, \ldots, k-b$ is optimal.

Properties 1 and *2* allow us to eliminate all arcs that start in node $x_k(n, i)$, for all $k = 1, 2, \ldots, m-1$ and $i = 1, 2, \ldots, n$, to all nodes in subsequent stages, if a node $x_{k+1}(n, i)$ exists. A maximum number of $n(n-1+n^2)(m-1)$ arcs could be reduced in this fashion. The effect of *Properties 1* and *2* is shown in Fig. 4 for the example system described in Fig. 3.

Property 3: Given a block of b vehicles that are assigned color i and consecutive positions $k+1, \ldots, k+b$, then pulling a vehicle l that has been assigned color $j \neq i$ and inserting it inside the block is never optimal.

Proof: Breaking a block of consecutive vehicles that have been assigned the same color i is never optimal, since the cost of changing over from color i would have to be incurred twice.

Normally, nodes of type $x_k(p, i, j)$ are connected by arcs to all nodes in stages $k+2, k+3, \ldots, m$. However, if it is possible to assign color j to vehicles in stages $k+2, k+3, \ldots, k+b$, then, all arcs from node $x_k(p, i, j)$ to any node in stages $k+1$ to $k+b$ can be eliminated. This could possibly reduce the number of arcs that originate from node $x_k(p, i, j)$ by as much as $(b-1)(n+n^2)$.

Property 4: Given a block of b vehicles that are assigned color i and are originally in positions $k+1, \ldots, k+b$, then pulling any vehicle from this block and inserting it somewhere else in the sequence does not improve the solution.

Proof: The changeover from color i will have to be incurred at least once. Removing any vehicle from the block and inserting it somewhere will not obviate this need.

Property 4 allows us to eliminate all nodes of type $x_k(p, i, i)$ for $k = 1, 2, \ldots, m-1$ and $i = 1, 2, \ldots, n$, since these nodes would correspond to the breaking of a block. By eliminating these nodes, we also eliminate all incoming and originating arcs. The effect of *Properties 3* and *4* on the size of the network is shown in Fig. 5.

Property 5: If a vehicle k is pulled and is assigned color i and vehicles $k+1, k+2, \ldots, k+b$ are assigned color j, then vehicle k should be inserted either: 1) immediately after the $k+1$

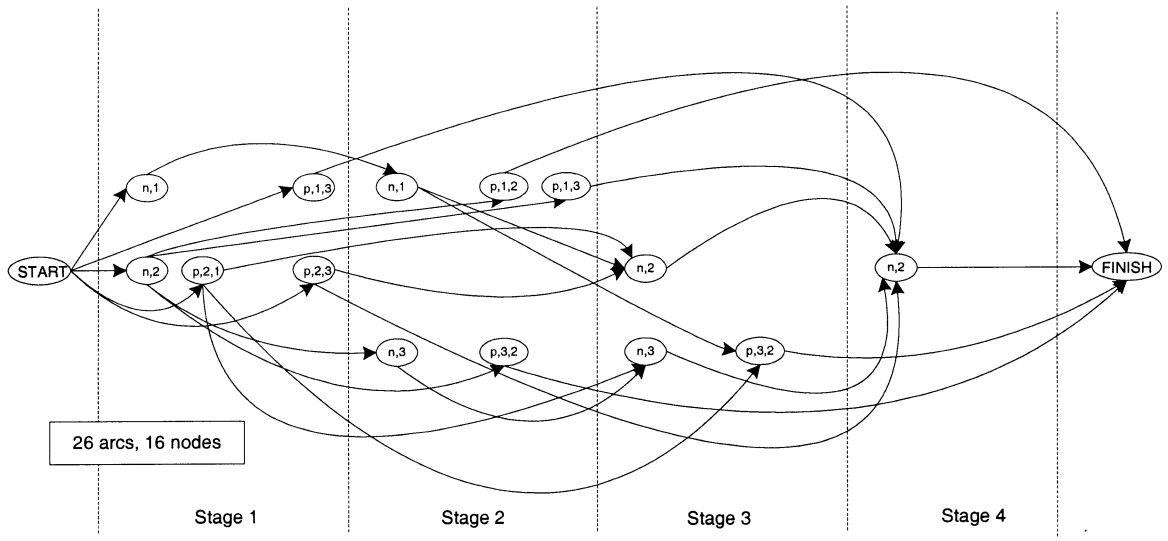


Fig. 6. Effect of applying *Property 5* on problem complexity.

to $k+b$ block; 2) immediately before another vehicle that has been assigned color i; or 3) one vehicle behind another vehicle that has been assigned color i. Inserting it anywhere else does not improve the solution.

Proof: The proof follows by noting that a vehicle k, which has been assigned color i, can only be pulled for one of two reasons. The first is to allow the formation of a block of two or more vehicles with color i downstream in the sequence. The second is to allow vehicles $k-1$ and $k+1$ to belong to the same block of identically colored vehicles. The first objective can be achieved either by placing vehicle k immediately behind another one with the same color or one vehicle behind the vehicle with the same color. In the latter case, the vehicle between the two can be pulled and inserted elsewhere. The second objective is satisfied by inserting vehicle k immediately after the block of j colored vehicles while guaranteeing that all other downstream sequences and color assignments will be considered.

Property 5 allows us to eliminate all arcs that originate from node $x_k(p, i, j)$ except those that have destinations in stage $k+b+1$ or those that have destinations in nodes $x_l(n, i)$ or $x_l(p, i', i)$ for $l > k+b+1$. The effect of *Property 5* on the size of the network is illustrated in Fig. 6. As we can see, applying *Properties 1–5* can significantly reduce the size of the network. In our example system, the number of arcs is reduced from 63 to 26, and the number of nodes from 19 to 16.

D. Integrated Solution

The solution to the problem with N pull-off tables is obtained by solving a series of N problems with one pull-off table. The final sequence obtained from each iteration serves as the original sequence in the next iteration. Although at each iteration, we solve both the resequencing and color-assignment problems, the color assignments made in iteration i are discarded in iteration $i+1$ as we resolve for a new sequence and a new color assignment. A solution is still obtainable in polynomial time since the complexity grows only linearly in N. Hence, the worst-case complexity of the overall procedure is order $O(Nm^2n^4)$. Note that the decomposition procedure does not guarantee optimality.

V. Numerical Results

We conducted numerical experiments to answer the following three questions.

1) How computationally efficient is our solution approach, and is it amenable to implementation in environments where we must obtain a solution in few seconds?

2) How good is the quality of the solution we obtain, and how is it affected by problem characteristics, such as number of vehicles, number of colors, number of pull-off tables, and density of the color-assignment matrix?

3) How much cost saving can be realized by introducing one or more pull-off tables and is it ever desirable to have full sequencing flexibility?

A. Computational Efficiency

To answer the first question regarding computational efficiency, we conducted a series of experiments for randomly generated problems with varying sizes, number of pull-off tables, and color density matrices. Color costs are sampled from a distribution that approximates the distribution of the costs in our application. A total of 52 000 examples were generated for problems with $m = 10$, 15, 20, 50, 100, and 200; $n = 10$, 15, and 20; $N = 0, 1, \ldots, 15$, and color densities ranging from 0.1 to 0.9. The color matrix density refers to the density of the matrix $[\mathbf{a}_{ij}]$. Matrix density, d, is calculated as the ratio of the sum of a_{ij}'s with a value of one over the total number of a_{ij}'s

$$d = \sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}/mn. \tag{16}$$

The value of d ranges from $1/n$ to 1, where the lower limit occurs if every vehicle can be painted with only one color, and the upper limit occurs if every vehicle can be painted with any color. Since matrices with similar densities could have different distributions of 1s and 0s, we generate 50 sample matrices for each density level and record the average CPU time (in milliseconds) for each level.

TABLE I
CPU TIMES (IN MILLISECONDS) FOR SYSTEMS WITH 15 COLORS

Number of Vehicles	Matrix Density	Number of pull-off tables									
		1	2	3	4	5	6	7	8	9	10
10	0.1	0	0	0	0	0	0	0	0	0	0
	0.2	0	0	0	1.6	1.6	3.2	3.2	4.7	4.7	4.7
	0.3	0	1.6	3.1	4.6	9.2	9.2	9.2	12.3	12.3	13.8
	0.4	3.2	4.8	6.4	9.6	12.7	17.3	17.3	23.4	24.9	26.5
	0.5	6.2	7.8	12.5	15.7	18.9	22	26.7	34.4	36	39.2
	0.6	1.5	7.8	13.9	18.7	21.7	26.4	34.3	37.5	37.5	46.9
	0.7	4.6	9.3	15.5	21.9	25	25	40.6	40.6	43.7	49.9
	0.8	3.1	7.8	15.7	17.3	22	26.7	26.7	34.6	37.7	39.3
	0.9	1.6	7.7	7.7	12.3	15.5	15.5	17.1	20.2	23.4	26.4
15	0.1	0	1.6	1.6	1.6	1.6	3.2	3.2	4.7	6.3	6.3
	0.2	4.7	7.9	7.9	9.5	9.5	12.7	15.8	19	19	22.1
	0.3	6	12.2	18.4	20	29.5	34.2	37.2	46.6	48.2	52.9
	0.4	12.5	28	34.2	46.8	62.4	67.1	79.6	91.9	101.4	112.4
	0.5	16.9	37.4	53	71.7	87.5	102.9	121.8	137.3	154.5	170.2
	0.6	21.9	45.3	65.6	92.3	114.1	131.2	159.4	178	201.7	221.8
	0.7	28	51.5	78.2	98.4	123.6	146.9	173.3	195.3	222	243.7
	0.8	21.7	43.6	67.1	84.4	106.2	125	146.6	165.5	187.4	207.6
	0.9	10.9	20.5	29.9	39.2	45.5	57.8	67.3	78.4	86.2	95.4
20	0.1	0	3.1	3.1	3.1	4.6	7.7	7.7	7.7	9.3	10.8
	0.2	6.2	9.4	12.6	17.3	22	26.6	29.7	32.8	37.5	43.8
	0.3	15.6	29.6	46.8	56.2	68.8	81.2	93.7	107.7	123.4	131.2
	0.4	28.2	54.6	86	107.7	137.4	164	189.1	217.1	239	267.3
	0.5	48.7	95.4	142.3	187.8	231.4	276.9	320.6	364.3	409.7	451.6
	0.6	62.8	120.6	182.9	242.3	306.4	367.4	425.2	484.6	544	601.9
	0.7	68.6	135.9	204.5	267.2	334.3	402.9	471.8	538.8	602.9	671.8
	0.8	62.6	121.7	182.8	241.9	303.1	365.4	426.5	485.8	545.2	607.6
	0.9	25.1	53	79.8	107.9	136	164.1	190.6	217	243.9	268.8
50	0.1	15.7	28.1	39	53.3	67	73.4	87.6	96.9	107.7	118.8
	0.2	62.5	115.6	174.8	229.6	289	346.9	403.2	457.7	513.7	566.8
	0.3	212.3	420.1	620.2	820.2	1015.5	1217.1	1410.9	1607.8	1807.7	1998.5
	0.4	512.6	1007.8	1509.5	2006.4	2495.4	2987.6	3468.8	3956.3	4434.5	4914.2
	0.5	956.1	1912.3	2859.1	3798.3	4732.6	5667.1	6600	7529.4	8460.8	9395.1
	0.6	1415.6	2842.1	4265.5	5700	7117.2	8542.2	9948.5	11349.9	12746.9	14117.1
	0.7	1776.6	3517.2	5284.2	7042.1	8757.8	10461	12142.1	13831.3	15490.6	17131.3
	0.8	1740.6	3468.7	5192.1	6911	8618.7	10333	12043.8	13756.3	15459.3	17159.3
	0.9	1118.7	2221.8	3298.4	4354.7	5403	6465.6	7521.8	8574.8	9628	10690.3

Illustrative examples of computational times for varying problem sizes and color density matrices are shown in Table I (the complete data set is available from the authors upon request). Solution for most realistic problems (i.e., 10-15 colors, 15-20 vehicles, and 1-5 pull-off tables) can be obtained within few seconds on a Pentium II workstation. For example, for problems with 50 vehicles, 15 colors, and 5 pull-off tables, a solution can be obtained in less than three seconds. However, even for large problems, a solution can be obtained within few minutes. For example, for systems with 200 vehicles, 20 colors, and 15 pull-off tables, a solution is obtained in less than seven minutes. As expected, computational times increase with increases in the number of vehicles, number of colors, or number of pull-off tables. Computational times are also sensitive to the density level. For either low or high density, computational times are relatively small. Computational times peak in the midrange, between 0.6 and 0.8, where the solution space tends to be the largest. This can be explained as follows. The solution space is limited at low densities, since few color options are available to each vehicle. It is also limited under high densities, since large blocks of similarly colored vehicles can be easily formed without resequencing.

In our application, the planning horizon, due to uncertainties in the upstream processes, is typically limited to 20 vehicles. However, our computational results indicate that longer planning horizons (e.g., over 100 vehicles) can be accommodated within the stated computational time constraint of one minute. This constraint is itself derived from the desire to resolve the problem with the passage of each vehicle, if necessary. However, this is usually needed only if a disruption occurs to the initial sequence or if color requirements change. If such a constraint is relaxed and a solution is required only once per planning period, then significantly longer planning horizons could be accommodated.

B. Solution Quality

To answer the question regarding solution quality, we first benchmarked the solution we obtained against a set of solv-

TABLE II
COMPARISONS BETWEEN OPTIMAL SOLUTIONS AND SOLUTIONS OBTAINED USING DECOMPOSITION ALGORITHM

Matrix Density	Number of pull-off tables					
	2			3		
	Algorithm	Optimal	% diff	Algorithm	Optimal	% diff
0.1	401.3	390.2	2.8	385.1	362.7	6.2
0.2	293.9	282.6	4.0	288.7	256.0	12.8
0.3	218.4	209.9	4.0	210.4	197.1	6.7
0.4	179.1	163.6	9.5	167.1	146.9	13.8
0.5	118.4	116.8	1.4	117.8	112.8	4.4
0.6	91.2	85.0	7.3	88.8	74.1	19.8
0.7	71.4	66.7	7.0	71.4	63.9	11.7
0.8	53.4	52.1	2.5	53.4	52.1	2.5
0.9	40.0	40.0	0.0	40.0	40.0	0.0

able problems with two and three pull-off tables. The size of the problems we considered (13 vehicles and 13 colors) is representative of realistic problems for our application. The problems are solved for densities of the color matrix ranging from 0.1 to 0.9. To obtain optimal solutions, we used the commercial software CPLEX version 6.5. The results are shown in Table II. As we can see, the difference in cost between the optimal solution and the solution obtained using our decomposition procedure ranges from 0 to 9.5% for systems with two pull-off tables with an average of 4.3%, and from 0 to 19.8% for systems with three pull-off tables with an average of 8.6%. The difference is especially small when density is high, which is again due to the fact that larger blocks are easier to identify and form with little or no resequencing.

We also benchmarked our results against an upper bound and a lower bound. The upper bound is given by the optimal solution to the problem with a single pull-off table, and the lower bound is given by the optimal solution to the problem when full flexibility is feasible. The lower bound is obtained by solving optimally the corresponding set-covering problem (the solution to this problem is generally time consuming and obviously not amenable to real-time implementation for large problems). We found that for the realistic cases of 15 vehicles or less and 10 or 15 colors, the difference between the two bounds does not exceed 25% with an average of 13.7%. Since only limited flexibility is usually available, the lower bound is relatively loose. Therefore, these results seem to indicate that our decomposition method would yield reasonably good solutions for problems with multiple pull-off tables. This is confirmed in Table III, where we compare the results obtained using our procedure against the lower bound. We can see, for example, that with three pull-off tables, the difference between our solution and the lower bound for the case of 15 vehicles and 10 colors has an average of 10.5%. This becomes significantly smaller with additional pull-off tables. The difference is also significantly smaller for systems with shorter resequencing horizons. For example, for systems with a 10-vehicle planning horizon, the difference from the lower bound is only 5% with just two pull-off tables.

C. Effect of Multiple Pull-Off Tables

To answer the third question, we examined the effect of introducing one or more pull-off tables on total cost. We first examined the benefits obtained from a single pull-off table by comparing the optimal solution we obtain with one pull-off table with the solution to the optimal color-assignment problem without resequencing. This benchmark is an enhanced version of current practice at the Ford factory, where colors are selected using a simple greedy algorithm which assigns the next vehicle the same color as the current one whenever feasible. Representative results are shown in Table IV for systems with 10 and 15 colors. Relative to this base benchmark, we can see that, for systems with 15 vehicles or less, up to 30% improvement can be achieved with a single pull-off table. The percentage improvement is higher for systems with a larger number of vehicles. The amount of improvement is sensitive to the color matrix density, with the largest improvements realized for low-to-medium densities. The effect of a pull-off table is insignificant for very high densities, since relatively large vehicle blocks of similar color can be formed through color assignment only.

To assess the impact of introducing one or more pull-off tables at the Ford plant, we collected data for 11 days of actual production. We obtained a sequence of approximately 12 000 vehicles. We then applied our algorithm sequentially to groups of 15 vehicles at a time, which corresponds to the anticipated resequencing horizon at the plant. We recorded the cost obtained per period using varying numbers of pull-off tables. Percentage daily improvements due to different numbers of pull-off tables are summarized in Fig. 7. The introduction of a single pull-off table results in an average cost reduction of 26%, or $57 per resequencing period. Assuming approximately 45 vehicles are produced per hour, this would result in annual savings of $1.2 million. The actual savings to Ford could be higher since we benchmarked our results against the optimal color assignment and not the myopic heuristic Ford currently uses.

We also studied the incremental benefits of additional pull-off tables. As we can see from Fig. 8, the effect of additional pull-off tables is of the diminishing kind with most of the benefits realized with one pull-off table. Increasing the number of pull-off tables beyond two or three tends to have only a marginal effect on performance. In the case of Ford (see Fig. 7), the introduction of a second pull-off table results in only an additional 3% in cost savings or $160 000 per year. Additional pull-off tables result in even smaller savings.

The impact of multiple pull-off tables can also be seen by considering Fig. 9. Here, we show the improvement due to the

TABLE III
COMPARISONS BETWEEN LOWER BOUND AND DECOMPOSITION ALGORITHM

Number of vehicles	Matrix density	Number of pull-off tables												Set covering
		1		2		3		4		5		6		
		Cost	% diff	Cost	% diff	Cost	% diff	Cost	% diff	Cost	% diff	Cost	% diff	
10	0.1	373.3	11.8	348.1	5.5	340.6	3.4	337.9	2.6	337.9	2.6	337.9	2.6	329.1
	0.2	301.6	11.7	283.5	6.1	276.6	3.7	273.8	2.7	272.6	2.3	272.1	2.1	266.4
	0.3	212.2	15.1	197.1	8.6	191.5	6.0	189.6	5.0	188.0	4.2	188.0	4.2	180.1
	0.4	159.9	15.3	148.3	8.6	144.0	5.9	143.5	5.6	143.0	5.2	143.0	5.2	135.5
	0.5	122.4	13.2	115.5	8.0	113.6	6.4	113.1	6.0	113.1	6.0	113.1	6.0	106.3
	0.6	95.8	9.8	91.4	5.4	91.4	5.4	91.4	5.4	91.3	5.4	91.3	5.4	86.4
	0.7	71.8	4.0	71.8	4.0	71.5	3.6	71.3	3.4	71.3	3.4	71.3	3.4	68.9
	0.8	60.6	0.8	60.6	0.8	60.6	0.8	60.5	0.6	60.5	0.6	60.5	0.6	60.1
	0.9	37.3	0.0	37.3	0.0	37.3	0.0	37.3	0.0	37.3	0.0	37.3	0.0	37.3
15	0.1	548.8	24.0	485.9	14.1	462.6	9.8	448.2	6.9	444.7	6.2	443.3	5.9	417.3
	0.2	414.9	23.8	370.8	14.7	350.8	9.9	341.3	7.4	338.6	6.6	333.0	5.0	316.2
	0.3	316.7	24.9	287.0	17.1	270.8	12.1	263.7	9.7	260.2	8.5	256.3	7.2	238.0
	0.4	241.0	26.5	218.8	19.1	208.4	15.0	200.0	11.5	194.6	9.0	193.6	8.6	177.1
	0.5	160.2	23.9	147.5	17.4	144.1	15.4	142.9	14.7	142.3	14.3	140.9	13.5	121.9
	0.6	124.7	18.6	117.7	13.7	115.2	11.9	114.0	11.0	112.3	9.6	112.0	9.4	101.5
	0.7	93.8	18.0	90.1	14.6	88.6	13.2	88.0	12.6	88.0	12.6	87.8	12.4	76.9
	0.8	68.0	7.4	67.9	7.3	67.7	7.0	67.6	6.9	67.6	6.9	67.3	6.5	62.9
	0.9	47.9	0.6	47.9	0.6	47.8	0.3	47.8	0.3	47.8	0.3	47.8	0.3	47.6
20	0.1	738.7	35.0	647.0	25.8	602.5	20.4	572.7	16.2	557.5	13.9	549.5	12.7	479.8
	0.2	538.3	32.8	476.0	24.0	445.4	18.8	430.4	16.0	419.7	13.8	412.1	12.2	361.7
	0.3	412.8	33.4	366.0	24.9	342.8	19.8	327.6	16.1	312.7	12.1	310.7	11.5	274.9
	0.4	300.5	33.4	265.7	24.6	250.7	20.1	240.2	16.6	235.9	15.1	233.3	14.1	200.3
	0.5	214.5	33.1	198.0	27.5	185.3	22.5	180.5	20.4	176.1	18.4	172.9	17.0	143.6
	0.6	155.4	28.1	143.3	22.0	139.0	19.6	135.2	17.3	134.1	16.6	132.3	15.5	111.8
	0.7	112.7	25.8	109.4	23.5	106.9	21.7	105.1	20.4	101.7	17.7	100.6	16.8	83.7
	0.8	82.6	12.1	79.8	9.0	79.2	8.4	79.2	8.4	79.2	8.4	79.2	8.4	72.6
	0.9	52.1	1.0	52.1	1.0	52.1	1.0	52.1	1.0	52.1	1.0	52.1	1.0	51.6
50	0.1	1738.7	64.2	1477.9	57.9	1334.9	53.4	1239.7	49.8	1178.1	47.2	1132.3	45.1	622.2
	0.2	1354.8	60.4	1161.5	53.8	1047.7	48.8	976.6	45.0	929.3	42.2	897.6	40.2	536.8
	0.3	983.4	57.8	846.3	51.0	773.0	46.4	730.8	43.3	697.7	40.6	674.4	38.5	414.5
	0.4	668.8	56.4	580.4	49.7	530.3	45.0	501.4	41.8	481.9	39.5	463.6	37.1	291.8
	0.5	493.9	57.4	435.3	51.7	401.1	47.6	380.4	44.8	366.0	42.6	352.9	40.4	210.2
	0.6	343.4	54.3	310.3	49.5	289.3	45.8	277.9	43.6	269.3	41.7	261.2	40.0	156.9
	0.7	242.1	51.7	218.0	46.3	209.2	44.1	202.5	42.2	196.2	40.4	194.5	39.8	117.0
	0.8	159.4	42.9	148.0	38.5	142.2	36.0	139.0	34.5	136.2	33.2	134.4	32.3	91.0
	0.9	83.2	24.8	79.7	21.5	79.7	21.5	79.7	21.5	79.2	21.0	79.2	21.0	62.5

number of pull-off tables as a fraction of the maximum possible improvement, which is realized when full sequencing flexibility is possible. We find that 65%–75% of the maximum possible cost reduction is realized with only one pull-off table, with the incremental benefit of additional tables quickly diminishing. This means that, in general, a limited number of pull-off tables would be sufficient, and that full resequencing would rarely be justified.

VI. SUMMARY AND DISCUSSION

In this paper, we presented a model and a solution procedure for job resequencing and feature assignment on a moving assembly line with limited resequencing flexibility. We showed that for a single pull-off table, our solution to the problem is optimal. For systems with multiple pull-off tables, we provided a decomposition approach, which we showed to result in good solutions for realistic problems. We found that the benefit of additional pull-off tables is of the diminishing kind, with most of the value of full resequencing flexibility achieved with only few pull-off tables. In our example application at Ford, we found that the implementation of our solution algorithm with the introduction of only one pull-off table would result in yearly savings in excess of $1.2 million. Furthermore, we showed that the value of resequencing is sensitive to the feature density matrix, with resequencing having little impact on cost when density is either high or low. The effect of resequencing can, however, be significant when density is in the middle range.

In this paper, we have focused on solving the vehicle resequencing and color-assignment problem for a fixed number of pull-off tables. However, our model and solution procedures could be used to identify the optimal number of pull-off tables when this is a decision variable. This requires trading off the marginal benefit of each additional pull-off table against the marginal cost of acquiring and operating an additional pull-off table. Since changeover costs are decreasing and convex in the number of pull-off tables, an optimal solution can be quickly identified in most cases. In the case of Ford, we found that an investment in a single pull-off table is optimal given an initial investment of $200 000 per pull-off table, an estimated operating

TABLE IV
IMPACT OF ONE PULL-OFF TABLE ON TOTAL COST

Matrix density	Systems with 10 colors			Systems with 15 colors		
	Optimal cost with no pull-off tables	Optimal cost with one pull-off table	Percentage improvement	Optimal cost with no pull-off tables	Optimal cost with one pull-off table	Percentage improvement
0.1	472.2	373.3	20.9	458.2	360.2	21.4
0.2	382.3	301.6	21.1	348.7	275.4	21.0
0.3	278.9	212.2	23.9	242.9	185.0	23.8
0.4	217.8	159.9	26.6	189.8	144.9	23.7
0.5	160.6	122.4	23.8	143.6	112.2	21.9
0.6	118.6	95.8	19.3	100.9	83.0	17.7
0.7	83.5	71.8	14.0	70.7	61.8	12.7
0.8	64.4	60.6	6.0	52.9	51.7	2.4
0.9	37.7	37.3	0.8	35.9	35.8	0.4
0.1	709.0	548.8	22.6	693.4	543.1	21.7
0.2	562.0	414.9	26.2	494.2	371.8	24.8
0.3	435.0	316.7	27.2	376.9	280.7	25.5
0.4	331.9	241.0	27.4	275.7	203.5	26.2
0.5	225.1	158.2	29.7	195.9	145.1	26.0
0.6	159.1	124.7	21.6	140.7	108.2	23.1
0.7	123.0	93.8	23.7	101.7	82.1	19.2
0.8	79.8	68.0	14.8	66.7	58.2	12.7
0.9	50.5	47.9	5.2	40.2	39.8	1.0
0.1	972.9	738.7	24.1	932.2	723.5	22.4
0.2	738.9	538.3	27.2	671.0	494.1	26.4
0.3	586.9	412.8	29.7	476.0	352.0	26.0
0.4	423.4	300.5	29.0	347.8	253.4	27.1
0.5	307.2	214.5	30.2	248.6	177.1	28.8
0.6	212.9	155.4	27.0	183.9	138.0	24.9
0.7	146.6	112.7	23.1	120.0	97.2	19.0
0.8	102.0	82.6	19.0	83.4	70.4	15.7
0.9	56.8	52.1	8.3	50.8	49.3	2.9
0.1	2405.7	1738.7	27.7	2326.8	1767.4	24.0
0.2	1915.4	1354.8	29.3	1652.5	1191.5	27.9
0.3	1421.9	983.4	30.8	1199.3	837.1	30.2
0.4	1011.7	668.8	33.9	839.7	570.6	32.1
0.5	751.4	493.9	34.3	594.8	403.4	32.2
0.6	524.0	343.4	34.5	428.4	293.6	31.5
0.7	363.1	242.1	33.3	299.0	208.5	30.3
0.8	223.4	159.4	28.7	191.6	135.5	29.3
0.9	106.0	83.2	21.5	97.8	74.0	24.3

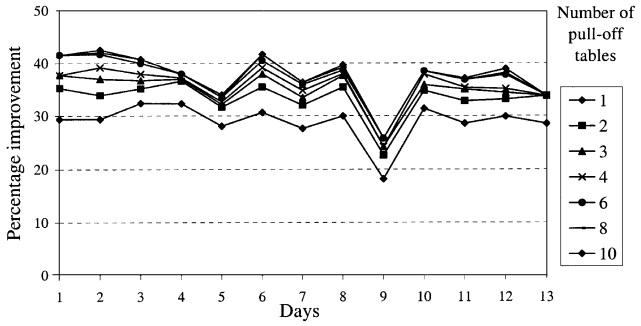


Fig. 7. Percentage daily improvements due to different number of pull-off tables.

cost of $120 000 per year, and a three-year payback period. This would result in annual net savings of approximately $1 million per paint line.

A number of future research avenues are possible. For example, in a multistage assembly line, a sequence that might be optimal for one stage is most likely to be suboptimal for others. Therefore, the benefits of resequencing at one stage should be appropriately traded off against losses at subsequent stages. This could be accomplished by extending the single-stage model to include multiple stages and the possibility of resequencing at every stage. In cases where only a limited number of offline buffers are available, there is also a need to develop procedures for allocating buffers among the various stages.

In certain applications, in addition to varying by cost, different features vary in processing time. In these cases, the need to form large blocks of jobs with the same features should be balanced against the need to have a leveled workload. Therefore, an alternative objective function that captures both the advantage of higher throughput as well as smaller changeover costs needs to be developed. Finally, instead of generating solutions

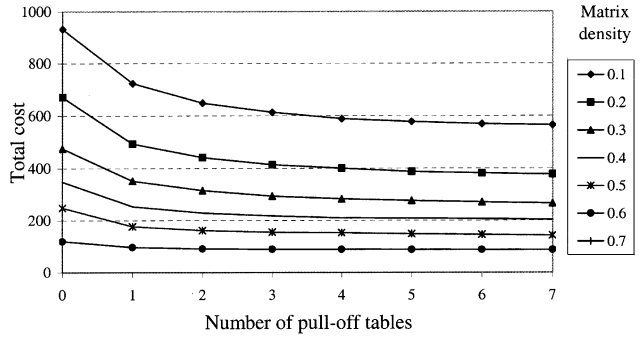


Fig. 8. Impact of multiple pull-off tables on total cost.

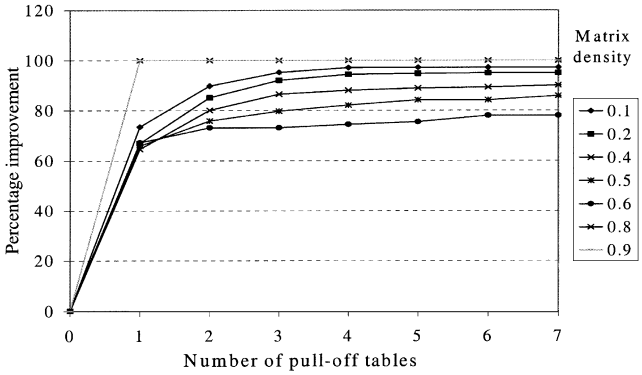


Fig. 9. Percentage improvement due to pull-off tables as a fraction of the maximum possible improvement.

for known blocks of jobs at a time, it might be desirable, in cases where the job sequence is subject to frequent disruption, to continuously revise the current solution each time a job is completed or a new job is added. This would require solving a slightly modified version of the current model where the offline buffers are not always initially empty.

APPENDIX I

Proposition A1: When full resequencing flexibility is possible, the resequencing and color-assignment problem reduces to a set-covering problem.

Proof: When full sequencing flexibility is feasible, the problem reduces to choosing a set of colors that results in the lowest cost, such that all vehicles are assigned one of these colors. Note that in this case, each color can occur only once in the final sequence, since multiple vehicle blocks of the same color could always be eliminated by resequencing. The problem is clearly equivalent to a set-covering problem of the following form:

$$\text{Minimize} \qquad z = \sum_{j=1}^{n} c_j x_j \tag{17}$$

Subject to:

$$\sum_{j=1}^{n} a_{i,j} x_j \geq 1 \qquad i = 1, 2, \ldots, m \tag{18}$$

where

$$x_j = \begin{cases} 1, & \text{if color } j \text{ is selected} \\ 0, & \text{otherwise.} \end{cases}$$

Although the above formulation does not explicitly provide us with a final sequence and vehicle color assignment, an optimal sequence and color assignment (generally, there are many optimal solutions) can be obtained using the following simple algorithm.

Step 1: Let the selected colors be denoted by arbitrary indices $j = 1, 2, \ldots, m^*$.

Step 2: Assign each vehicle to one of the selected colors.

Step 3: Let n_i denote the number of vehicles assigned to color i. Assign these vehicles positions $(n_1+n_2+\cdots+n_{i-1}+1)$ to n_i. Note that the ordering of the color blocks, as well as of vehicles within a block, is not important due to the full resequencing flexibility we have.

APPENDIX II

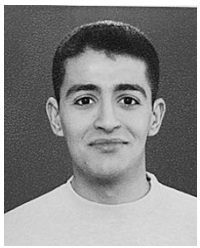
Proposition A2: The maximum total number of arcs that originate from nodes of type $x_k(p, i, j)$ is $n^3(m-1)(m-2)/2 + n^4(m-2)(m-3)/2$.

Proof: From nodes $x_k(p, i, j)$, there are no feasible arcs to any nodes in stage $k+1$, since if vehicle k is pulled, it cannot be inserted before a vehicle in stage $k+1$. However, there are feasible arcs from nodes $x_k(p, i, j)$ to all the nodes of type $x_{k'}(n, i')$ in subsequent stages. The total number of such arcs in the network is given by $\sum_{i=1}^{m-1} n^2 n(m-i-1) = n^3(m-1)(m-2)/2$. In addition, there are feasible arcs from nodes $x_k(p, i, j)$ to all the nodes of type $x_{k'}(p, i', j')$ in subsequent stages, with the exception of stage $m-1$. Therefore, the total number of such arcs is given by $\sum_{i=1}^{m-3} n^2 n^2 (m-i-2) = n^4(m-2)(m-3)/2$. Summing both terms, we obtain $n^3(m-1)(m-2) + n^4(m-2)(m-3)/2$.

REFERENCES

[1] D. L. Myron and T. L. Magnanti, "Paint blocking in Ford's in-line vehicle sequencing environment," Leaders for Manufacturing Program, Cambridge, MA, Tech. Rep., 1996.
[2] I. Baybars, "A survey of exact algorithms for the simple assembly line balancing problem," *Manage. Sci.*, vol. 32, pp. 902–952, 1986.
[3] C. A. Yano and A. Bolat, "Survey, development, and application of algorithms for sequencing paced assembly lines," *J. Manuf. Oper. Manage.*, vol. 2, pp. 172–198, 1989.
[4] C. Y. Lee and G. Vairaktarakis, "Workforce planning in mixed model assembly systems," *Oper. Res.*, vol. 45, pp. 553–567, 1997.
[5] C. A. Yano and R. Rachamadugu, "Sequencing to minimize work overload in assembly lines with product options," *Manage. Sci.*, vol. 37, no. 5, pp. 572–586, 1991.
[6] L. D. Burns and C. F. Daganzo, "Assembly line job sequencing principles," *Int. J. Prod. Res.*, vol. 25, pp. 71–99, 1987.
[7] A. Bolat, M. Savsar, and M. A. Al-Fawzan, "Algorithms for real-time scheduling of jobs on mixed model assembly lines," *Comput. Oper. Res.*, vol. 21, pp. 487–498, 1994.
[8] C. L. Monma and C. N. Potts, "On the complexity of scheduling with batch setup times," *Oper. Res.*, vol. 37, pp. 798–804, 1999.
[9] C. N. Potts and L. N. Van Wassenhove, "Integrating scheduling with batching and lot sizing: A review of algorithms and complexity," *J. Oper. Res. Soc.*, vol. 43, pp. 395–406, 1992.
[10] A. T. Unal and A. S. Kiran, "Batch sequencing," *IIE Trans.*, vol. 24, pp. 73–83, 1992.
[11] S. Webster and K. R. Baker, "Scheduling groups of jobs on a single machine," *Oper. Res.*, vol. 43, pp. 692–703, 1995.

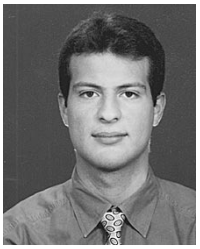
[12] J. Bruno and P. Downey, "Complexity of task sequencing with deadlines, setup times, and changeover costs," *SIAM J. Comput.*, vol. 7, pp. 393–404, 1978.
[13] G. Laporte, "The traveling salesman problem: an overview of exact and approximate algorithms," *Eur. J. Oper. Res.*, vol. 59, pp. 231–247, 1992.
[14] E. Balas and M. C. Carrera, "A dynamic subgradient-based branch and bound procedure for set covering," *Oper. Res.*, vol. 44, pp. 875–890, 1996.
[15] A. Caprara, M. Fischetti, and P. Toth, "A heuristic method for the set covering problem," *Oper. Res.*, vol. 47, pp. 730–743, 1999.
[16] M. Lahmar and S. Benjaafar, "Resequencing with limited flexibility," Grad. Pgm. in Ind. Eng., Univ. of Minnesota, Minneapolis, MN, Working Paper, 2003.



Maher Lahmar received the B.S. and M.S. degrees in industrial engineering from Bilkent University, Ankara, Turkey.

He is currently working toward the Ph.D. degree in industrial engineering at the University of Minnesota, Minneapolis. His research interests are in the areas of manufacturing systems, production and inventory control, and facility planning.

Mr. Lahmar served as President of the IIE Chapter, University of Minnesota, Minneapolis, in 2000–2001 and is a member of IIE and INFORMS since 1998.



Hakan Ergan received the B.S. degree in industrial engineering from Bosphurus University, Istanbul, Turkey, and the M.S. degree in industrial engineering from the University of Minnesota, Minneapolis.

He is currently working in the Operations Research Group, USAirways, Pittsburgh, PA.

Mr. Ergan is a member of IIE and INFORMS.

Saif Benjaafar (M'92) received the Ph.D. and M.S. degrees in industrial engineering from Purdue University, West Lafayette, IN, and the B.S. degree in electrical and computer engineering from the University of Texas at Austin.

He is a Professor in the Department of Mechanical Engineering, University of Minnesota, Minneapolis, where he is Director of the Industrial Engineering Division and the Center for Supply Chain Research. His research interests are in manufacturing and service operations and supply chain management. His papers have appeared in several journals including *IIE Transactions*, IEEE TRANSACTIONS ON ROBOTICS AND AUTOMATION, *Management Science*, and *Interfaces*.

Dr. Benjaafar is Associate Editor for the *Journal of Manufacturing Systems* and the *International Journal of Flexible Manufacturing Systems*. He is a recipient of the IIE Outstanding Young Industrial Engineer Award, the IIE Transactions Best Paper Award in 1999 and 2000, the SME Ralph E. Cross Outstanding Young Engineer Award, and the National Science Foundation Research Initiation Award.