FACULTY WORKING PAPER 93-0106

Product Assignment in Flexible Multilines
Part 1: Single-Stage Systems with Demand Splitting

Narayan Raman
Department of Business Administration
University of Illinois

Udatta S. Palekar
Department of Mechanical and Industrial Engineering
University of Illinois

Bureau of Economic and Business Research
College of Commerce and Business Administration
University of Illinois at Urbana-Champaign

BEBR

FACULTY WORKING PAPER NO. 93-0106

College of Commerce and Business Administration

University of Illinois at Urbana-Champaign

January 1993

Product Assignment in Flexible Multilines
Part 1: Single-Stage Systems with Demand Splitting

Narayan Raman

Udatta S. Palekar

Product Assignment in Flexible Multilines
Part 1: Single-Stage Systems with Demand Splitting

Narayan Raman

Department of Business Administration

University of Illinois at Urbana-Champaign

Champaign, Illinois

Udatta S. Palekar

Department of Mechanical and Industrial Engineering

University of Illinois at Urbana-Champaign

Urbana, Illinois

January 1993

ABSTRACT

This study addresses the design of a serial manufacturing system with multiple parallel lines. Such systems manufacture a variety of products in medium to large volumes with stable demand rates and similar processing requirements. Each line comprises multiple identical machines which perform a set of predetermined, product-specific tasks on the products assigned to that line. Given the fixed cost of providing a line, and the fixed cost of each machine, the objective of the flexible multiline design problem is to determine the product-to-line assignment that minimizes the total investment in lines and workcenters. We consider the special case of a single-stage system in which a product can be assigned to multiple lines. While this case arises as an important subproblem in the general multi-stage problem, it merits independent consideration in many systems in which the *same* stage is the bottleneck for all products. We develop important characteristics of the optimal solution; in particular, we show that it must satisfy the *sequential assignment property* which renders it solvable in polynomial time. We develop an efficient enumerative solution method that makes effective use of an imbedded greedy algorithm.

This study considers the problem of designing a flexible multi-line in a serial manufacturing system. Such systems process a variety of products in medium to large volumes with stable demand rates. These products have similar processing requirements in that they visit the various manufacturing stages in the same sequence. Each stage on any line comprises multiple identical machines which perform a set of predetermined tasks on the products assigned to that line. While these tasks require similar processing capabilities, the actual tasks done and their processing times are product-specific. The flexible machines can switch from one product to another with negligible changeover time. The adjacent stages are tightly coupled with minimal buffer storage space in between. Each line is paced, and therefore, its cycle time is constrained by the maximum processing time across all stages required by any product assigned to it.

Given the fixed cost of providing a line, and the fixed cost of each workcenter at each stage, the objective of the flexible multiline design (FMD) problem is to partition the set of products such that each subset is assigned to exactly one line, and the total investment in lines and workcenters is minimized. The FMD problem arises in several manufacturing environments. Consider, for example, the manufacturing facility of a Midwestern company in heavy engineering industry. In this facility, large parts are painted on several parallel paint lines. The parts are suspended on overhead conveyors and transported through pretreatment, painting and drying stages. The time required at each stage varies for different products depending upon their size and shape but the conveyors are paced. In another instance, a manufacturing company in the auto industry that produces fuel-supply systems, requires different components to go through a series of forming operations at the fabrication stage. These components are processed on parallel lines that are paced by the longest processing time of any product on that line. The FMD problem is also encountered in printed circuit board manufacture (Farber, Hanan and Luss 1988). Indeed, this problem arises naturally in many systems in the context of implementing a just-in-time approach within cellular manufacture. Given a set of products with their individual demands and processing requirements, FMD determines the optimal set of families, as well as the optimal configuration of the various cells that need to be formed. Additionally, it can be used at periodic intervals to evaluate the need for a system redesign in the face of changing product demands and processing needs.

The problem most closely related to the FMD problem is the mixed-model line balancing problem (Wester and Kilbridge 1964; Thomopolous 1967, 1970; MacAskill 1972; Dar-El 1978; Okamura and Yamashita 1979; and Yano and Rachamadugu 1991). Much of the previous work on mixed-model line balancing problem addresses the assignment of tasks required for assembling a number of products to operators stationed along an assembly line. In the basic model, the operators are considered to be multiskilled so that the tasks can be assigned to any operator on the line as long as the precedence relations among them are satisfied. It is easy to see that in such tandem systems, the cycle time and the overall output are constrained by the total processing time required at the bottleneck station. Consequently, the bulk of the research on this problem has considered the objective of smoothing workload assignments across all stations. Because of the variety of products assembled on this line, the amount of processing required at any station varies from one cycle to another, and workload balancing is based on the average processing time per cycle at each station. Work overloads are relieved by permitting limited operator movement upstream and downstream of the assigned station (Dar-El and Cucuy 1977, Dar-El 1978), or through the use of utility workers (Yano and Rachamadugu 1991). One of the major thrusts of this research is on determining the appropriate sequence in which the various models should be processed at each station in order to minimize such overloads. Okamura and Yamashita (1979) address the objective of minimizing the maximum distance that any worker will have to move away from his workstation in order to complete all tasks assigned to him; as Yano and Rachamadugu (1991) note, this objective is similar to minimizing the maximum work overload at any station. Yano and Rachamadugu deal with the objective of minimizing the average work overload given that the overload at any station can be met through the use of utility workers.

An alternative line of research involving mixed-model lines addresses sequencing the various products with the objective of smoothing the rate of parts usage in assembling the final products. This problem was proposed by Monden (1983) in the context of just-in-time manufacture. Miltenberg (1989) considers the problem in which all final products require the same number and mix of parts. Under this assumption, smoothing part usage rate reduces to minimizing the sum of differences between the cumulative actual production and cumulative actual demands across all products. Miltenberg proposes nonlinear integer programming formulations, and proposes heuristic solution methods. Kubiak and Sethi (1991) relax Miltenberg's assumption, and also consider a more general form of the objective function; more importantly, they show that the resulting problem can be formulated as an assignment problem. Similar problems are studied by Miltenberg and Sinnamon (1989) and Inman and Bulfin (1991).

The FMD problem is similar to mixed-model line balancing in that it considers a paced flow line producing multiple products. In addition, the objective of minimizing total investment in lines and workcenters leads to workload balancing. However, these two problems differ in significant ways. First, the assignment of tasks to stations (stages) is not an issue here because any given task can be done only at a predetermined stage. Second, the stages are "manned" by *stationary* CNC machines. Consequently, there can be no variation in the time spent at any station from one cycle to another, and the sequence in which the different models are run is immaterial. Workload balance in our context is achieved purely by the formation of parallel lines and grouping products with similar processing times on a line. While there are economic incentives in having multiple lines in order to reduce idle time, the benefits of doing so need to be traded off against the fixed cost of providing the lines.

Another problem related to the multiline design problem is the line segmentation problem (LSP) considered by Ahmadi and Matsuo (1991). For a given number of machines at each stage, and a given partition of products into families such that each family is assigned to one line, the objective of LSP is to allocate machines at each stage to individual lines such that the overall makespan is minimized. Ahmadi and Matsuo present several heuristics for solving LSP and show their efficacy with respect to valid lower bounds. The FMD problem differs from LSP in two important ways. First, LSP considers a *multi-model* situation in which the entire (daily) demand of any product is produced in one batch before the line changes over to produce the next product. In our *mixed-model* approach, each product is allowed to be produced as often as desired subject to the overall demand constraints. Second, FMD addresses a problem in which product-to-line allocation is done jointly with the determination of the number of lines and the number of workstations required at each stage for each line.

This paper is the first of two papers that together address our research on the FMD problem. In both papers, we consider the special case in which there is only one stage. This special case arises as an important subproblem while solving the general multi-stage problem. However, this case merits independent consideration for many systems in which the *same* stage is the bottleneck for all products; for such systems, the multi-stage FMD problem reduces to a single-stage problem. Furthermore, there are several systems that have only one stage. The two papers differ in that this paper considers overlapping product partitions; the demand of any product can then be spread across several lines. In the companion paper (Palekar and Raman 1993), we address the case in which each product is constrained to be produced on only one line.

This paper is organized as follows. The problem formulation is given in §1. We develop some dominance properties in §2 that result in an efficient graph representation of the FMD problem. This representation is used in §3 to generate the optimal solution based on a dynamic programming approach. We also develop an alternative polynomial-time algorithm that makes repeated use of a greedy heuristic algorithm. We conclude in §4 with a summary of the main results of this paper.

1 Problem Description

In this section, we present a mixed integer programming formulation of the flexible multiline design problem. However, first we give the notation used in the paper.

$\mathcal{N}$ = the set of products, and $|\mathcal{N}| = N$

F_1 = fixed cost of opening a line

F_2 = fixed cost per machine

p_j = processing time of product j, $j \in \mathcal{N}$

$\mathcal{J}_i$ = set of products with processing times greater than or equal to i, $\{j | p_j \geq p_i, \ j \in \mathcal{N}\}$

d_j = per period demand of product j, $j \in \mathcal{N}$

A = available time per period on any machine

τ_l = cycle time of line l

$\mathcal{L}_l$ = the set of products assigned to line l

In any feasible solution, the cycle time τ_l of line l equals the processing time of its *pivot* product $\pi(l)$, i.e., the product with the longest processing time that is assigned to that line. Also, for any pivot j, let $\lambda(j)$ denote the index of the corresponding line, and n_j denote the number of workcenters required at this line $\lambda(j)$. Then, the cycle time of any line l with pivot $\pi(l) = j$ is $\tau_l = p_j$ and its capacity is An_j/p_j. We assume that $A \gg p_j, \ \forall j \in \mathcal{N}$ so that $\lfloor A/p_j \rfloor \approx A/p_j$. We also assume that the fixed costs F_1 and F_2 are nonnegative integers.

The flexible multiline design problem is stated as

FMD1

$$\text{Minimize} \quad Z = \sum_{j=1}^{N} (F_1 y_j + F_2 n_j) \tag{1}$$

subject to

$$\sum_{j \in \mathcal{J}_i} x_{ji} = 1, \quad i \in \mathcal{N} \tag{2}$$

$$p_j \left(\sum_{i \in \mathcal{N}} d_i x_{ji} \right) \leq A n_j, \quad j \in \mathcal{N} \tag{3}$$

$$x_{ji} \leq y_j, \quad i, j \in \mathcal{N} \tag{4}$$

$$x_{ji} \geq 0, \quad i, j \in \mathcal{N} \tag{5}$$

$$y_j \in \{0, 1\}; n_j \geq 0, \text{ integer}, \quad j \in \mathcal{N} \tag{6}$$

where

$$y_j = \begin{cases} 1, & \text{if a line is opened with pivot } j \\ 0, & \text{otherwise} \end{cases}$$

and x_{ji} is the fraction of product i's demand assigned to line $\lambda(j)$ if such a line exists, otherwise, it is zero. Equation (2) insures that the demand of each product is fully assigned, and a product is assigned only to lines with cycle times no less than the processing time of the product. Constraint (3) requires that all product-to-line assignments be capacity feasible. Constraint (4) insures that the fixed cost of opening a line is accounted for. Finally, constraints (5) and (6) specify the nature of the variables.

The total number of machines required on any line l with pivot j is

$$n_j = \left\lceil \frac{\sum_{i \in \mathcal{N}} d_i x_{ji} p_j}{A} \right\rceil,$$

where $\lceil f \rceil$ is the smallest integer greater than or equal to f. The total idle time on line l is

$$A n_j - p_j \left(\sum_{i \in \mathcal{N}} d_i x_{ji} \right).$$

Clearly the idle time on this line is reduced by assigning to it products which have processing times close to p_j, and therefore, **FMD1** aims at balancing processing times. On the other hand, the mixed-model line balancing problem aims at balancing workloads. Furthermore, the idle time on any line in **FMD1** is unaffected by the sequence in which the various products are processed.

2 Problem Representation

In this section, we develop dominance properties, and construct an efficient graph representation of problem **FMD1**.

Proposition 1. *There exists an optimal solution to* **FMD1** *with pivot set* $\mathcal{P} = \{j | j \in \mathcal{N}, y_j = 1\}$ *such that* $p_k \neq p_l$ *for* $k, l \in \mathcal{P}$, $k \neq l$.

PROOF: For any optimal solution σ to **FMD1** that does not have the above property, we construct an alternative solution σ' from σ by merging line $\lambda(k)$ with line $\lambda(l)$ while the assignments on other lines remain unchanged. Then

$$\begin{aligned} Z(\sigma) - Z(\sigma') &= 2F_1 + F_2 \left\{ \left\lceil \frac{p_l \sum_{t\in\mathcal{N}} d_t x_{tl}}{A} \right\rceil + \left\lceil \frac{p_k \sum_{t\in\mathcal{N}} d_t x_{tk}}{A} \right\rceil \right\} \\ &\quad - F_1 - F_2 \left\{ \left\lceil \frac{p_l (\sum_{t\in\mathcal{N}} d_t x_{tl} + \sum_{t\in\mathcal{N}} d_t x_{tk})}{A} \right\rceil \right\} \\ &\geq 0, \end{aligned}$$

where the inequality follows from $F_1 \geq 0$, $p_k = p_l$, and the inequality

$$\lceil a + b \rceil \leq \lceil a \rceil + \lceil b \rceil. \tag{7}$$

Hence, if σ is optimal, then so is σ' and the proof is complete. □

Proposition 2. *There exists an optimal solution to* **FMD1** *with pivot set* $\mathcal{P} = \{j | j \in \mathcal{N}, y_j = 1\}$ *such that*

i) if i is not a pivot product, then it is assigned to exactly one line, i.e., $x_{ui} \in \{0, 1\}$ *for all* $i \in \mathcal{N} \backslash \mathcal{P}$ *and* $u \in \mathcal{P} \cap \mathcal{J}_i$.

ii) if i is a pivot product, then it is assigned to at most two lines.

PROOF: As before, we show that any solution σ that is optimal to **FMD1** and that does not have the stated property can be modified to yield an alternative optimal solution that does so. Without loss of generality, we assume that σ satisfies Proposition 1. Let L be the total number of lines in σ, and let $\mathcal{L}_l$ denote the set of products assigned to line l, $l = 1, 2, \ldots, L$. Renumber these lines so that

$$\tau_1 > \tau_2 > \ldots > \tau_L. \tag{8}$$

Let $D_l = \sum_{i \in \mathcal{N}} d_i x_{\pi(l)i}$ denote the total quantity assigned to line l, $l = 1, 2, \ldots, L$. Construct another solution σ' from σ in the following manner. Rank all products in $\mathcal{N}$ in nonincreasing order of their processing times. Starting from line 1, assign products from the top of this list, such that the total quantity assigned to line l is D_l. If this results in any product being partially assigned to a line, then allocate the remaining quantity to the subsequent line. Let τ'_l and $\mathcal{L}'_l$, respectively, denote the cycle time of line l and the set of products assigned to line l, $l = 1, 2, \ldots, L$ in σ'.

Note that σ' satisfies condition i) in that any product which is not a pivot is assigned to exactly one line. Also note that

$$\tau'_1 = \tau_1. \tag{9}$$

Lemma 1. $\tau'_l \leq \tau_l, \forall l.$

PROOF: Consider the following disjunctive cases:

Case a) $\underline{min_{i \in \mathcal{L}'_l}\{p_i\} \leq \tau_{l+1}, \quad l = 1, 2, \ldots, L-1}$: From the construction of σ', it can be seen that $\tau'_{l+1} \leq min_{i \in \mathcal{L}'_l}\{p_i\} \leq \tau_{l+1}$, for $l = 1, 2, \ldots, L-1$. The result follows from (9).

Case b) $\underline{min_{i \in \mathcal{L}'_l}\{p_i\} > \tau_{l+1}, \quad \text{for some } \ l \in \{1, 2, \ldots, L-1\}}$: Because $\tau_l = max_{q \in \mathcal{L}_l}\{p_q\}$ for any line l, it follows from (8) that

$$\tau_{l+1} \geq p_i, \quad \forall i \in \bigcup_{k=l+1}^{L} \mathcal{L}_k.$$

Furthermore, since $min_{i \in \mathcal{L}'_l}\{p_i\}$ is strictly larger than τ_{l+1}, and the total quantity allocated to each line is the same in both σ and σ', it follows that

$$\bigcup_{k=1}^{l} \mathcal{L}_k = \bigcup_{k=1}^{l} \mathcal{L}'_k.$$

This implies that $\tau'_{l+1} = \tau_{l+1}$ to yield the desired result. □

Now

$$Z(\sigma') - Z(\sigma) = \sum_{i \in \mathcal{N}} \sum_{l=1}^{L} \lceil \tau'_l D_l / A \rceil - \sum_{i \in \mathcal{N}} \sum_{l=1}^{N} \lceil \tau_L d_l / A \rceil \leq 0 \tag{10}$$

and σ' is optimal. If it satisfies property ii) as well, the proof is complete. Otherwise, merge all those lines that have the same pivot to construct another solution σ'' that satisfies both i) and ii) and, from Proposition 1, is optimal as well. □

In the rest of this paper, we assume without any loss of generality that the products are numbered such that if $i < j$, then $p_i \geq p_j$, for any $i, j \in \mathcal{N}$. Furthermore, since $F_1 \geq 0$, they satisfy Proposition 1 which can now be restated as

Remark 1. *In any optimal solution, $p_k > p_l$ for any $k, l \in \mathcal{P}$ such that $k > l$.*

Proposition 3. *(Sequential Assignment Property) There exists an optimal solution to* **FMD1** *with the property that if $x_{ji} > 0$, then $x_{jq} = 1$ for $q = j+1, j+2, \ldots, i-1$.*

PROOF: Let σ be an optimal solution to **FMD1** that does not have this property. Then there exists at least one product $t, j < t \leq i-1$ that is produced on line $\lambda(k), k \neq j$. Note that $k \leq t < i$. If $k < j$, then construct σ' from σ by shifting $\delta = d_t x_{kt}$ units of product t from line $\lambda(k)$ to line $\lambda(j)$. Replace these units on line $\lambda(k)$ by products, considered in the increasing order of their index starting with j, that are currently assigned to line $\lambda(j)$. If $\delta < d_j x_{jj}$, then j continues to remain a pivot, otherwise its entire demand is absorbed by line $\lambda(k)$ and j is replaced as a pivot by some q, with $p_q \leq p_j$. In either case, $Z(\sigma') \leq Z(\sigma)$, and therefore, σ' is optimal.

If $k > j$, then construct σ' by shifting $\delta = \sum_{q=t+1}^{i} d_q$ units of demand corresponding to products $t+1$ through i from line $\lambda(j)$ to line $\lambda(k)$, and replace these units on line $\lambda(j)$ with products currently assigned to line $\lambda(k)$ considered in the increasing order of their index starting with k. As before, it follows that $Z(\sigma') \leq Z(\sigma)$, and therefore, σ' is optimal. Repeating these steps whenever required yields the solution σ' that is optimal to **FMD1** and that satisfies the condition stated in the proposition. □

Hereafter, we deal only with those solutions that satisfy the sequential assignment property. An immediate consequence of the above propositions is that in an optimal solution, if j is a pivot in an optimal solution, then it is assigned to at most two lines, namely $\lambda(j) - 1$ and $\lambda(j)$, i.e., $x_{qj} > 0$, only if $q \in \{\pi(\lambda(j) - 1), j\}$. Furthermore, $x_{ju} = 1$ for all u, $u = j+1, j+2, \ldots, \pi(\lambda(j)+1) - 1$. In addition, an optimal solution is completely characterized by the set of pivot products in the solution.

Problem **FMD1** can be represented on graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ shown in Figure 1. In this graph, node V_{ij}, which is depicted as ij in the figure, represents an assignment in which product j is produced on line with pivot i. Note that node V_{ij} is feasible only if $p_j \leq p_i$; hence, the upper triangular nature of this graph. Let E_{ij}^{uv} denote the arc leading from V_{ij} to V_{uv}. The arcs in $\mathcal{G}$ connect only contiguous nodes, i.e., an arc E_{ij}^{uv} exists only if $v = j + 1$. [We will show later that each arc joining two nodes in Figure 1 actually represents a set of arcs.] We append a dummy source node S and a dummy sink node T. The optimal solution to **FMD1** corresponds to the shortest path from S to T.

INSERT FIGURE 1 HERE

We partition arc-set $\mathcal{E}$ into disjoint subsets $\mathcal{H}$, $\mathcal{B}$ and $\mathcal{F}$ where $\mathcal{H}$ is the set of *horizontal* arcs *(h-arcs)*, $\mathcal{B}$ is the set of *backward* arcs *(b-arcs)*, and $\mathcal{F}$ comprises the *forward* arcs *(f-arcs)*. Consider an arc $E_{ij}^{u,j+1} \in \mathcal{E}$. $E_{ij}^{u,j+1}$ is an h-arc if $u = i$; a b-arc if $u < i$; and an f-arc otherwise. Without any loss of generality, we assume that the dummy arcs leading to S and T are f-arcs.

From the sequential assignment property (SAP), it follows that any path that includes a b-arc is not dominant. Furthermore, in an optimal solution, a pivot must be assigned to its own line, since otherwise, the products assigned to this line are being produced at a higher than required cycle time. Consequently, if V_{ij} lies in an optimal path, then so must V_{ii}. Together with SAP, this implies that V_{ij} is reachable only via node V_{ii} along the path comprising the h-arcs $E_{ii}^{i,i+1} - E_{i,i+1}^{i,i+2} - \ldots - E_{i,j-1}^{ij}$. Therefore, while searching for an optimal solution, we need consider only those f-arcs that are incident on a pivot, i.e., arcs of the form $E_{ij}^{j+1,j+1}$. Notice that Figure 1 does not have any other f-arcs.

Clearly, product 1 must be the pivot for line 1 in any feasible solution. Consider a path Ω that passes through $V_{11}, \ldots, V_{1,i-1}, \ldots$. Corresponding to node V_{jk}, let M_{jk} denote the number of machines required on line $\lambda(j)$ for producing products $j, j+1, \ldots, k$. Then the number of machines required at line 1 in Ω is

$$n_1 = M_{1,i-1} = \left\lceil \frac{p_1 \sum_{u=1}^{i-1} d_u}{A} \right\rceil .$$

The capacity remaining, hereafter the *remnant*, at line 1 after this assignment is

$$r_{1,i-1} = \frac{AM_{1,i-1}}{p_1} - \sum_{u=1}^{i-1} d_u .$$

It is clearly optimal to use $r_{1,i-1}$ for (partially) meeting the demand of product i. More generally, the remnant available at any line will be used for producing the pivot product of the next line. Consequently, if i and j are adjacent pivots in Ω, $j > i$, then

$$M_{jk} = \left\lceil \frac{p_j \left(\sum_{u=j}^{k} d_u - r_{i,j-1} \right)}{A} \right\rceil \tag{11}$$

and

$$r_{jk} = \frac{AM_{jk}}{p_j} - \left(\sum_{u=j}^{k} d_u - r_{i,j-1} \right). \tag{12}$$

Note that $r_{jk} < A/p_j$, hence, it is strictly less than one machine's capacity on line $\lambda(j)$. Since $p_j \geq p_k$, for $k > j$, it is less than one machine's capacity on subsequent lines as well.

We now determine the cost of each arc in $\mathcal{G}$. First note that the cost of f-arc $E_{jk}^{k+1,k+1}$ leading to node $V_{k+1,k+1}$, $k = 0, 1, \ldots, N-1$ is

$$c_{jk}^{k+1,k+1} = F_1 + F_2 M_{k+1,k+1} = F_1 + F_2 \left\lceil \frac{p_{k+1}\left(d_{k+1} - r_{jk}\right)}{A} \right\rceil. \tag{13}$$

where $k = 0$ denotes the source node S. The remnant at $V_{k+1,k+1}$ is

$$r_{k+1,k+1} = \frac{AM_{k+1,k+1}}{p_{k+1}} - d_{k+1} + r_{jk}. \tag{14}$$

From (12), it follows that r_{jk} and therefore, $c_{jk}^{k+1,k+1}$ and $r_{k+1,k+1}$ as well, depend upon i, the pivot for the line that immediately precedes $\lambda(j)$. By induction this implies that they depend upon the path selected to reach V_{jk}. We assume that the cost of dummy f-arcs leading to T is zero.

Now consider h-arc $E_{jk}^{j,k+1}$. The marginal increase in the number of machines required on line $\lambda(j)$ for producing $(k+1)$, given that products $j+1, j+2, \ldots, k$ are assigned to this line, is

$$\begin{aligned} M_{j,k+1} - M_{jk} &= \left\lceil \frac{p_j \left(\sum_{u=j}^{k+1} d_u - r_{i,j-1} \right)}{A} \right\rceil - \left\lceil \frac{p_j \left(\sum_{u=j}^{k} d_u - r_{i,j-1} \right)}{A} \right\rceil \\ &= \left\lceil \frac{p_j \left(d_{k+1} - r_{jk}\right)}{A} \right\rceil, \end{aligned}$$

where i is the pivot for the line immediately preceding $\lambda(j)$. The cost of h-arc $E_{jk}^{j,k+1}$ is

$$c_{jk}^{j,k+1} = F_2 \left\lceil \frac{p_j \left(d_{k+1} - r_{jk}\right)}{A} \right\rceil. \tag{15}$$

Arguments similar to those used above show that $c_{jk}^{j,k+1}$, and the remnant $r_{j,k+1}$ as well, depends upon the path selected to reach V_{jj}.

However, an exponential growth in the number of paths to be considered for reaching any node is avoided because of cost- and remnant-dominance. Consider two paths Ω_1 and Ω_2 from V_{11} to

V_{jk}. Let the resulting remnant at V_{jk} following Ω_1 and Ω_2 be r^1_{jk} and r^2_{jk}, respectively, and let $\mathcal{C}(\cdot)$ denote the cost of path $(\cdot)$. Then

Remark 2. *Ω_1 dominates Ω_2 if either*

i) $r^1_{jk} \geq r^2_{jk}$ and $\mathcal{C}(\Omega_1) \leq \mathcal{C}(\Omega_2)$, or

ii) $\mathcal{C}(\Omega_1) \leq \mathcal{C}(\Omega_2) - F_2$.

PROOF: Consider i) first. The optimal completion of Ω_1 solves a subproblem of **FMD1** that considers products $k+1, k+2, \ldots, N$ with demands $d_{k+1} - r^1_{jk}, d_{k+2}, \ldots, d_N$, respectively. On the other hand, the optimal completion of Ω_2 considers a subproblem in which the demand of product $k+1$ is $d_{k+1} - r^2_{jk} \geq d_{k+1} - r^1_{jk}$, demands of other products and other parameters remaining unchanged. Clearly, the optimal completion of Ω_2 is at least as expensive as the optimal completion of path Ω_1, and i) follows from the principle of optimality. Now consider ii). Note that $r^1_{jk}, r^2_{jk} < A/p_j$. Hence, from (13) and (15), the remnant difference $r^1_{jk} - r^2_{jk}$ cannot result in saving more than one machine. ii) follows immediately. □

From Remark 2, it follows that the set of undominated paths reaching $V_{k+1,k+1}$ from V_{jk} comprises only those paths whose costs are within F_2 of each other. Furthermore, these paths must have distinct r_{jk} values. Because there can be mo more than A/p_j such values, it follows that the cardinality of this set is no more than A/p_j.

Similar arguments show that when all paths reaching $V_{k+1,k+1}$ from V_{lk}, $l = 1, 2, \ldots, k-1$, are considered, then the set of undominated paths $\Psi_{k+1,k+1}$ reaching $V_{k+1,k+1}$ comprises only those paths that are within F_2 of each other in cost but have distinct $r_{k+1,k+1}$ values. Because there can be no more that A/p_{k+1} such values, $\psi_{k+1,k+1} = |\Psi_{k+1,k+1}| \leq A/p_{k+1}$. More generally, for any node V_{jj}, $j \in \mathcal{N}$, we have

$$\psi_{jj} = |\Psi_{jj}| \leq \frac{A}{min_{i \in \mathcal{N}}\{p_i\}} \stackrel{\text{def}}{=} \psi.$$

In summary,

Lemma 2. *Each f-arc and h-arc shown in Figure 1 represents a set of arcs whose cardinality is less than or equal to ψ and whose costs differ by less than F_2.*

In view of the above result, wherever appropriate, we will use the notation $c^{u,k+1}_{jk}(s)$ to denote the cost of arc $E^{u,k+1}_{jk}$ given that $r_{jj} = s$, $s = 1, 2, \ldots, \psi$.

3 Heuristic and Exact Solution Algorithms

FMD1 can be formulated as a dynamic program in which the stages are the consecutively numbered products. The states at a given stage i are given by the combination of the pivots, j, $j \leq i$, to which product i can be assigned, and the remnant s available at the pivot. Let $g^*(k)$ be the minimum cost of reaching stage k, $k = 1, 2, \ldots, T$, and $g_{jk}(s)$ be the minimum cost of reaching node V_{jk} given that the $r_{jj} = s$. The resulting shortest path problem is stated as

DP1

$$Z^* = min \quad g^*(T)$$

subject to

$$g^*(k) = min_{\substack{j \leq k \\ s \leq \psi}} \{g_{jk}(s)\} \tag{16}$$

$$g_{jk}(s) \quad = \quad \begin{cases} g_{j,k-1}(s) + c^{jt}_{j,k-1}(s), & \text{if } j < k \\ g_{i_s,k-1} + c^{jk}_{i_s,k-1}, & \text{if } j = k \end{cases} \tag{17}$$

$$g_{00}(s) = 0 \quad s = 1, 2, \ldots, \psi \tag{18}$$

$$i_s = arg \ min_{\substack{i<k \\ u \leq \psi}} \left\{ g^u_{i,k-1} + c^{kk}_{i,k-1} | r_{kk} = s \right\}. \tag{19}$$

Equation (19) insures that among all paths leading to a pivot node V_{kk}, $k = 1, 2, \ldots, N$ that result in the same remnant at V_{kk}, the path that is selected is one with the lowest cost.

DP1 requires evaluating $O(\psi N)$ arcs at each stage, resulting in an overall computational effort of $O(\psi N^2)$. We now present an alternative, enumerative solution approach for solving **FMD1**. This is done for two reasons. First, it is likely to be more efficient computationally for most real problems. Note that while **DP1** solves **FMD1** in polynomial time, ψ can be a large number. Second, the proposed algorithm makes use of a greedy method that we use extensively in the companion paper for solving the related problem **FMD2** in which demand splitting is not permitted. We now present this heuristic method.

3.1 A Greedy Algorithm

Consider a greedy policy in which the path leading to any pivot node V_{jj} is selected myopically on the basis of the total cost incurred in reaching that node. Remnant differences are used only to break ties in favor of the path that results in the largest remnant at V_{jj}. Clearly, because of its Markovian property, this policy considers only a subgraph $\mathcal{G}^G$ of $\mathcal{G}$ in which a pair of adjacent nodes is connected by only one h-arc or f-arc. Suppressing all references to remnants, we can state the resulting shortest path problem under the greedy policy as

DP2

$$Z^G = min \ g^G(T)$$

subject to

$$g^G(t) = min_{j \leq t}\{g^G_{jt}\} \tag{20}$$

$$g^G_{jt} = \begin{cases} g^G_{j,t-1} + c^{jt}_{j,t-1}, & \text{if } j < t \\ g^G_{h_{(t-1)},t-1} + c^{jt}_{h_{(t-1)},t-1}, & \text{if } j = t \end{cases} \tag{21}$$

$$h_{(t-1)} = arg \ max_{i<t}\left\{r_{i,t-1} | g_{i,t-1} + c^{jt}_{i,t-1} = min_{l<t}\left\{g_{l,t-1} + c^{jt}_{l,t-1}\right\}\right\}. \tag{22}$$

Equation (22) implements the tie breaking rule. Following (14), the f-arc selected to reach node V_{tt} is from node $V_{h_{(t-1)},t-1}$ that has the largest remnant among all nodes at stage $t-1$ which are on the shortest path from V_{11} to V_{tt}. **DP2** can be solved in $O(N^2)$ computation time. Hereafter, we refer to this solution method as **Greedy**. Because $\mathcal{G}^G$ considers only a subset of arcs in $\mathcal{E}$, the **Greedy** solution value is only an upper bound on the optimal solution value of **FMD1**. However, the following result indicates that these two solution values differ by no more than the cost of one machine.

Proposition 4. *Let Z^* be the optimal solution value to* **FMD1**, *and let Z^G denote the solution value of* **Greedy**. *Then, $Z^G < Z^* + F_2$.*

PROOF: Let Ω^* be an optimal path between V_{11} and T, and let $\mathcal{P}^* = \{\pi(1), \pi(2), \ldots, \pi(L)\}$ denote the set of pivots on Ω^*. Without any loss of generality, we assume that Ω^* satisfies the sequential assignment property. Let g^*_{jt} denote the cost of reaching any node V_{jt} on Ω^* from V_{11}. Then,

Lemma 3. *$g^G_{jk} < g^*_{jk} + F_2$, for all $k \in \mathcal{L}_{\lambda(j)}$, and $j \in \mathcal{P}^*$.*

PROOF: In order to distinguish the variables under $\mathcal{G}^G$ and Ω^*, we use $[\mathcal{G}^G]$ and $[\Omega^*]$, and wherever it does not result in any confusion, superscripts 'G' and '*', respectively. Let Ω^G denote the path followed by the **Greedy** solution in $\mathcal{G}^G$. For expositional simplicity, we assume that Ω^G is unique.

If Ω^* is included in $\mathcal{G}^G$, then Ω^* and Ω^G are identical, and the result follows trivially. Otherwise, there must exist a pivot $q \in \mathcal{P}^*$ such that the f-arc leading to node V_{qq} differs from the f-arc selected in $\mathcal{G}^G$ according to the greedy policy. Furthermore, since there is only one f-arc leading to node V_{22} in $\mathcal{G}$, it must be true that $q \geq 3$. Let V_{tt} be the first pivot node where the f-arcs differ. Then

$$g_{jt}^G = g_{jt}^*, \quad 1 \leq j \leq t-1, \;\; t \in \mathcal{L}_{\lambda(j)} \tag{23}$$

and the result holds for $j \leq t-1$. From (21) and (22), we have

$$g_{tt}^G \leq g_{tt}^*. \tag{24}$$

As shown in Figure 2, let the f-arc selected for reaching V_{tt} in $\mathcal{G}^G$ be $E_{u,t-1}^{tt}$ that results in a remnant of r_{tt}^G at V_{tt}. Similarly, let the f-arc selected for reaching V_{tt} in Ω^* be $E_{v,t-1}^{tt}$, resulting in a remnant of r_{tt}^* at V_{tt}. If $r_{tt}^* < r_{tt}^G$, then we can construct another path Ω' from V_{11} to T that is identical to Ω^* except in that the path segment selected for reaching node V_{tt} from V_{11} is replaced by the segment used in Ω^G. From Remark 2, it follows that if Ω' is optimal. If $r_{qq}^* < r_{qq}^G$ for all pivot nodes V_{qq} on Ω^*, the repeating the above substitution eventually shows that Ω^G is optimal, and the lemma is proved.

INSERT FIGURE 2 HERE

Now suppose that $r_{tt}^* > r_{tt}^G$. From the sequential assignment property, Ω^* must next visit either $V_{t+1,t+1}$ or $V_{t,t+1}$. In the following, we consider the case in which Ω^* passes through $V_{t+1,t+1}$. The proof for the other case is similar and it is, therefore, omitted. First note that the result holds for $j = t$. Now,

$$A/p_{t+1} > r_{tt}^* > r_{tt}^G. \tag{25}$$

From (11) and (25), it follows that either

$$i) \;\; M_{t+1,t+1}^* = M_{t+1,t+1}^G - 1, \tag{26}$$

$$\text{or } ii) \;\; M_{t+1,t+1}^* = M_{t+1,t+1}^G. \tag{27}$$

In i) a machine is saved at $V_{t+1,t+1}$ following Ω^* relative to $\mathcal{G}^G$. Hence,

$$c_{tt}^{t+1,t+1}\left[\mathcal{G}^G\right] = c_{tt}^{t+1,t+1}\left[\Omega^*\right] + F_2. \tag{28}$$

From (24), and (28), it follows that

$$g^G_{t+1,t+1} \leq g^G_{tt} + c_{tt}^{t+1,t+1}\,[\mathcal{G}^G] < g^*_{tt} + c_{tt}^{t+1,t+1}\,[\Omega^*] + F_2 = g^*_{t+1,t+1} + F_2 \tag{29}$$

and from (14), (25) and (26), that

$$r^G_{t+1,t+1} > r^*_{t+1,t+1}. \tag{30}$$

Ω^* must next visit $V_{w,t+2}$ where $w = t+1$, or $w = t+2$. In either case, from (13), (15) and (30), we have

$$c^{w,t+2}_{t+1,t+1}\,(\mathcal{G}^G) \leq c^{w,t+2}_{t+1,t+1}\,(\Omega^*). \tag{31}$$

Therefore, from (29) and (31)

$$g^G_{w,t+2} \leq g^*_{w,t+2} + F_2. \tag{32}$$

It follows that if $r^G_{iq} \geq r^*_{iq}$ for any node V_{iq} on Ω^*, $i \geq w$, then $g^G_{j,q+1} \leq g^*_{j,q+1} + F_2$ where either $i = j$, or i and j are consecutive pivots. If this condition holds for all nodes on Ω^*, then the lemma is true by induction. But $r^G_{iq} < r^*_{iq}$ at any node V_{jq} is possible only if a machine is saved at that node in $\mathcal{G}^G$ with respect to Ω^*. In that case, $g^G_{iq} \leq g^*_{iq}$, and again the result holds.

Now suppose that a machine is not saved at $V_{t+1,t+1}$ as in case ii). In this case, we have

$$c_{tt}^{t+1,t+1}\,[\mathcal{G}^G] = c_{tt}^{t+1,t+1}\,[\Omega^*].$$

Hence,

$$g^G_{t+1,t+1} \leq g^G_{tt} + c_{tt}^{t+1,t+1}\,[\mathcal{G}^G] < g^*_{tt} + c_{tt}^{t+1,t+1}\,[\Omega^*] = g^*_{t+1,t+1}$$

and $r^G_{t+1,t+1} < r^*_{t+1,t+1}$. As above, it spawns two cases at node $V_{w,t+2}$, and using similar arguments, it can be shown that (32) holds in this case as well. If the remnant difference $r^*_{t+1,t+1} - r^G_{t+1,t+1}$ does not save a machine at any subsequent node on Ω^*, then the result is true by induction once again. On the other hand, if a machine is saved at a subsequent node, then considering cases i) and ii) at that node and repeating the arguments made above completes the proof of the lemma. □

The result stated in the proposition follows immediately from Lemma 3 by substituting $j = \pi(L)$, and $k = N$, and noting that the cost of any f-arc leading to node T is zero. □

3.2 An Exact Algorithm

We construct an exact algorithm for **FMD1** that combines Lemma 3 with the **Greedy** solution. This method is based on an enumerative approach that considers $\mathcal{G}^G$ initially, and augments it with additional arcs from $\mathcal{G}$ whenever necessary. Suppose that the optimal solution to **FMD1** is represented by path Ω^* having pivots $j_1^*, j_2^*, \ldots, j_L^*$, with solution value Z^*. The proposed algorithm identifies j_L^* at level 1, and generates the lower numbered pivots in the nondecreasing order of their indices at the succeeding levels in the enumeration tree.

In the following, $\Gamma_j = \left\{V_{ij} | g_{ij}^G < g^G(j) + F_2\right\}$ denotes the set of nodes at stage j, $j = 1, 2, \ldots, N$, that can be reached from V_{11} at a cost within F_2 of the minimum cost of reaching stage j in $\mathcal{G}^G$. Also, let m_{jk} denote the (possibly fractional) number of machines required at line $\lambda(j)$ corresponding to node V_{jk}; i.e.,

$$m_{jk} = \frac{p_j \left(\sum_{u=j}^{k} d_u - r_{i,j-1}\right)}{A} \quad \text{and} \tag{33}$$

$$M_{jk} = \lceil m_{jk} \rceil . \tag{34}$$

Let $m_{jk} = a_{jk} + b_{jk}$ where a_{jk} is the integer part of m_{jk}. $b_{jk} = m_{jk} - \lfloor m_{jk} \rfloor$ is the purely fractional part of m_{jk}, where $\lfloor f \rfloor$ is the largest integer less than or equal to f. Also, let α_j and β_j denote, respectively, the integer and the purely fractional part of the number of machines required at line $\lambda(j)$, i.e., $n_j = \lceil \alpha_j + \beta_j \rceil$. Then, corresponding to node V_{jk}, $\alpha_j = a_{jk}$ and $\beta_j = b_{jk}$.

The proposed algorithm is formally stated below.

Algorithm **Exact**

Step 1: *Initial Solution and Pre-Processing*

i) Solve **FMD1** using **Greedy** with solution value Z^G. Set the current upper bound $UB = Z^G$. Record the root vertex as the current incumbent. For $j = 2, 3, \ldots, N$, and $i \leq j$, determine r_{ij} and compute

$$\Delta_{ij} = r_{ij} - r_{h_j j}; \quad \text{and} \quad \kappa_{ij} = g_{ij}^G - g_{h_j j}^G$$

where h_j is defined by (22). Determine Γ_N. Go to Step 2.

Step 2: *Branching, Updating and Fathoming*

Construct a search tree $\mathcal{S}$ rooted at T by generating a vertex at level 1 in $\mathcal{S}$ corresponding to each node in Γ_N. For each such vertex w that corresponds to (say) node V_{jN} in $\mathcal{G}^G$, set

$$\theta_v = b_{jN}, \quad \delta_v = p_j / A, \quad \text{and} \quad \phi_v = g_{jN}^G.$$

Determine Γ_{j-1} and generate vertices at the second level corresponding to nodes in Γ_j, and similarly generate vertices at other levels in $\mathcal{S}$ such that the descendants of any unfathomed vertex u, which corresponds to (say) node $V_{k+1,l}$, are vertices corresponding to nodes in Γ_k. For any vertex v, which is a descendant of vertex u and corresponds to (say) node V_{ik} in $\mathcal{G}^G$, set $\delta_v = \delta_u$, and $z_v = \Delta_{ik}\delta_v$.

If $z_v \geq \theta_u$, then set $\phi_v = g^G_{h_k k} + \kappa_{ik} - F_2$. If $\phi_v = Z^G - F_2 + 1$, then record v as the incumbent, and go to Step 3. Else if $\phi_v < UB$, then record v as the incumbent. Compute $\theta_v = 1 - (z_v - \theta_u)$.

Otherwise, if $z_v < \theta_u$, then set $\phi_v = g^G_{h_k k} + \kappa_{ik}$. Fathom v if $\phi_v \geq UB + F_2$. Else, set $\theta_v = \theta_u - z_v$.

If no further vertex can be generated, go to Step 3.

Step 3: *Generation of the Optimal Solution*

Let v be the incumbent vertex, which corresponds to (say) node V_{iq}. Trace the path leading from v to the root vertex in $\mathcal{S}$, and find the nodes in G corresponding to each vertex in this path. These nodes determine the corresponding path in $\mathcal{G}^G$ from V_{iq} to T. Find the shortest path from V_{11} to V_{iq} using **Greedy** to complete the solution.

Proposition 5. *Algorithm* **Exact** *solves* **FMD1**.

PROOF: Note that in order to determine Ω^*, it is sufficient to find the pivots $j_1^*, j_2^*, \ldots, j_L^*$. Alternatively, it suffices to identify the f-arcs leading to these pivots. Algorithm **Exact** generates these f-arcs in reverse starting with the f-arc leading to pivot j_L^*. We first show that the Ω^* is included in $\mathcal{S}$, and then show the validity of the costs determined for each vertex in $\mathcal{S}$.

Lemma 4. *There exists an optimal solution Ω' to* **FMD1** *with pivots $j_1', j_2', \ldots, j_Q'$, such that i) $g^G_{j_Q' N} < g^G(N) + F_2$, and ii) $g^G_{ik} < g^G(k) + F_2$, where i and $k+1$ are two consecutive pivots in Ω'.*

PROOF: From Lemma 3, we have

$$Z^* \leq Z^G = g^G(N) \leq g^G_{j_Q' N} < Z^* + F_2. \tag{35}$$

i) follows immediately. We will show that if Ω' does not also satisfy ii), then it can be modified to yield another optimal solution that does so. Clearly, if ii) does not hold, then there must be a node V_{tk} such that

$$g^G_{ik} \geq g^G(k) + F_2 = g^G_{tk} + F_2. \tag{36}$$

Modify Ω' by substituting pivot t for i; this may result in changing all pivots that precede i as well. Keep pivots $k+1$ through j'_Q unchanged. Then, it follows from (36), and the fact that the segment from $V_{k+1,k+1}$ to $V_{j'_Q N}$ is unchanged, that the modified solution is optimal as well. Repeating this step whenever ii) is violated eventually results in an optimal solution that satisfies ii). □

We assume in the following that Ω^* satisfies Lemma 4. Because we enumerate all nodes in Γ_N at level 1, Lemma 4i) implies that $V_{j^*_L N}$ is considered at this level. Furthermore, in the general case, since the algorithm generates vertices corresponding to all nodes in Γ_k after having generated a pivot $k+1$, from Lemma 4ii) it follows that Ω^* in included in $\mathcal{S}$.

Next we address the costs determined at each vertex. Consider the general case described in Step 2 of the algorithm and depicted in Figures 3a and 3b. For any two consecutive pivots, i and $k+1$ on any path Ω, let v denote the vertex in $\mathcal{S}$ corresponding to node $V_{ik} \in \Gamma_k$, u be the parent of v, and w be the vertex at level 1 from which u and v are generated. Let u and w correspond to nodes $V_{k+1,l}$ and V_{jN} in $\mathcal{G}^G$, respectively.

INSERT FIGURES 3a AND 3b HERE

Consider the descendants generated corresponding to Γ_k. From Equation (22), the minimum cost path leading from V_{11} to $V_{k+1,k+1}$ passes through $V_{h_k k}$. Then, $\kappa_{ik} = g^G_{ik} - g^G_{h_k k}$ is the cost penalty incurred if f-arc $E^{k+1,k+1}_{ik}$, instead of $E^{k+1,k+1}_{h_k k}$, is selected for reaching $V_{k+1,k+1}$. However, such a substitution in the f-arc can lead to the saving of a machine if the remnant difference $\Delta_{ik} = r_{ik} - r_{h_k k}$ absorbs the purely fractional part of a machine required at line $\lambda(j)$, i.e., the last line on Ω. To see this, let θ_a denote the value of the purely fractional machine to be eliminated from the last line corresponding to vertex a. Corresponding to w at the first level of the tree, we have $\theta_w = \beta_j = b_{jN}$. If we select the f-arc from $V_{t,j-1}$ to reach V_{jj}, instead of the f-arc from $V_{h_{j-1},j-1}$ selected in $\mathcal{G}^G$, then a cost penalty of $\kappa_{t,j-1} = g^G_{t,j-1} - g^G_{h_{j-1},j-1}$ is incurred.

Let τ denote the vertex in tree $\mathcal{S}$ corresponding to $V_{t,j-1}$, and let the variable values modified because of this f-arc switch be denoted by a 'prime'. If the remnant difference $\Delta_{t,j-1} = r_{t,j-1} - r_{h_{j-1},j-1}$ is such that

$$\Delta_{t,j-1}\frac{p_j}{A} \geq b_{jN}, \tag{37}$$

then from (33) and (34), $M'_{jN} = M_{jN} - 1$, and a machine is saved at line $\lambda(j)$. At vertex τ, we then have a solution value $\phi_\tau = g^G_{jN} + \kappa_{t,j-1} - F_2$, and the modified value of the purely fractional machine at $\lambda(j)$ of $\theta_\tau = \beta'_j = 1 - (\Delta_{t,j-1}p_j/A - f_{jN})$. On the other hand, if $\Delta_{t,j-1}$ does not satisfy (37), then no machine is saved at $\lambda(j)$. The f-arc switch results in a solution value $\phi_\tau = g^G_{jN} + \kappa_{t,j-1}$, and $\theta_\tau = \beta'_j = f_{jN} - (\Delta_{t,j-1}p_j/A$.

[Note that even if a machine saving is realized at τ, additional savings may still be possible at any of its descendants at succeeding levels.] It is easily seen that in the general case considered in Step 2 of the algorithm, if $\Delta_{ik}p_j/A \geq \theta_u$ at vertex v, then $\phi_v = g^G_{h_k k} + \kappa_{ik} - F_2$, and $\theta_v = 1 - (\Delta_{ik}p_j/A - \theta_u)$. On the other hand, if $\Delta_{ik}p_j/A < \theta_u$, then $\phi_v = g^G_{h_k k} + \kappa_{ik}$, and $\theta_v = \theta_u - \Delta_{ik}p_j/A$.

Finally, from Lemma 3, it follows that a vertex v is fathomed if its solution value $\phi_v \geq UB + F_2$ since any completion of v can be no better than the incumbent solution. Furthermore, because F_1 and F_2 are integers, $LB = Z^G - F_2 + 1$ is a lower bound on Z^*. Therefore, if $\phi_v = LB$ at any node v, then search can be terminated. □

3.3 An Example Problem

We illustrate the above algorithm with the following 5-product example $F_1 = 50; F_2 = 100; A = 800; d_1 = 60; d_2 = 280; d_3 = 180; d_4 = 1000; d_5 = 1900; p_1 = 5.0; p_2 = 2.5; p_3 = 2.0; p_4 = 1.0; p_5 = 0.45$. At the end of step 1, the **Greedy** solution is $V_{11} - V_{22} - V_{23} - V_{44} - V_{55}$ with a value $Z^G = 800$. The g^G_{jt} and κ_{jt} values are shown in Table 1, while the r_{jt} and the Δ_{jt} values are given in Table 2.

INSERT TABLES 1 AND 2 HERE

From Table 1, it can be seen that $\Gamma_5 = \{V_{45}, V_{55}\}; \Gamma_4 = \{V_{44}\}; \Gamma_3 = \{V_{13}, V_{23}, V_{33}\}; \Gamma_2 = \{V_{11}, V_{22}\}; \Gamma_1 = \{V_{11}\}$.

The enumeration tree is shown in Figure 4, and the details for this tree are given in Table 3. Paths corresponding to the **Greedy** and the optimal solutions are shown on graph $\mathcal{G}^G$ in Figure 5. The optimal path is $V_{11} - V_{22} - V_{33} - V_{44} - V_{55}$ with a value $Z^* = 750$. The arcs shared by both paths are shown with double lines while the arcs exclusive to the optimal path are shown in thin bold lines.

INSERT TABLE 3, AND FIGURES 4 AND 5 HERE

4 Conclusion

This paper addresses the flexible multi-line design problem in a single-stage manufacturing system. For a given fixed cost of providing a line, and the fixed cost of each workcenter, the objective of the flexible multi-line design problem is to simultaneously determine the number of lines required as well as find the product-to-line allocation such that the total investment in lines and workcenters is minimized.

In this paper, we consider the case in which a product can be assigned to multiple lines. We show that in this case, the optimal solution must satisfy the *sequential assignment property*, i.e., the products assigned to any line must be consecutively ordered in their processing times. We give a dynamic programming algorithm that solves the problem in polynomial time. We develop an efficient enumerative solution method that makes effective use of an imbedded greedy algorithm.

REFERENCES

1. Ahmadi, R. H. and H. Matsuo (1991), "The Line Segmentation Problem," *Operations Research*, Vol. 39, 42–55.

2. Dar-El E. M. (1978), "Mixed-Model Assembly Line Sequencing Problems," *Omega*, Vol. 6, 317–323.

3. Dar-El, E. M. and S. Cucuy (1977), "Optimal Mixed-Model Sequencing for Balanced Assembly Lines," *Omega*, Vol. 5, 333- -341.

4. Farber, M., H. Luss and C.-S. Yu (1988), "Assembly Line Design Tools Line Balancing and Line Layout," Working Paper, AT&T Bell Laboratories, Holmdel, NJ.

5. Inman, R. and R. Bulfin (1991), "Sequencing JIT Mixed-Model Assembly Lines," *Management Science*, Vol. 37, 901–904.

6. Kubiak, W. and S. P. Sethi (1991), "A Note on "Level Schedules for Mixed-Model Assembly Lines in Just-in-Time Production Systems"," *Management Science*, Vol. 37, 121–122.

7. MacAskill, J. L. C. (1972), "Production Line Balances for Mixed Model Lines," *Management Science*, Vol. 19, 423–434.

8. Miltenberg, J. (1989), "Level Schedules for Mixed-Model Assembly Lines in Just-in-Time Production Systems," *Management Science*, Vol. 35, 192–207.

9. Miltenberg, J. and G. Sinnamon (1989), "Scheduling Mixed-Model, Multilevel Assembly Lines in Just-in-Time Production Systems," *International Journal of Production Research*, Vol. 27.

10. Monden, Y. (1983), *Toyota Production System*, Industrial Engineering and Management Press, Institute of Industrial Engineers, Atlanta, GA.

11. Okamura, K. and H. Yamashita (1979), "A Heuristic Algorithm for the Assembly Line Model-Mix Sequencing Problem to Minimize the Risk of Stopping the Conveyor, " *International Journal of Production Research*, Vol. 17, 233-247.

12. Palekar, U. S. and N. Raman (1993), "Product Assignment in Flexible Multilines, Part 2: Single Stage Systems with No Demand Splitting," Working Paper # 92–0120, Bureau of Economic and Business Research, Univ. of Illinois at Urbana-Champaign, IL.

13. Thomopolous, N. T. (1967), "Line Balancing-Sequencing for Mixed Model Assembly," *Management Science*, Vol. 14, 69–75.

14. Thomopolous, N. T. (1970), "Mixed Model Line Balancing with Smoothed Station Assignments", *Management Science*, Vol. 16, 593–603.

15. Wester, L. and M. Kilbridge (1964), "The Assembly Line Mixed Model Sequencing Problem," in *Proceedings of the Third International Conference on Operations Research*, Paris, France.

16. Yano, C. A. and R. V. Rachamadugu (1991), "Sequencing to Minimize Work Overload in Assembly Lines with Product Options," *Management Science*, Vol. 37, 572–586.

TABLE 1

Values of g_{jt}^G and κ_{jt} in the Example Problem

j	g_{jt}^G at $t =$					κ_{jt} at $t =$				
	1	*2*	*3*	*4*	*5*	*1*	*2*	*3*	*4*	*5*
1	150	350	450	1050	2250	0	50	50	500	1450
2	-	300	400	700	1300	-	0	0	150	500
3	-	-	450	650	1150	-	-	50	100	350
4	-	-	-	550	850	-	-	-	0	50
5	-	-	-	-	800	-	-	-	-	0
$g^G(t)$	150	300	400	550	800					
h_t	1	2	2	4	5					

TABLE 2

Values of r_{jt} and Δ_{jt} in the Example Problem

j	r_{jt} at $t =$					Δ_{jt} at $t =$				
	1	*2*	*3*	*4*	*5*	*1*	*2*	*3*	*4*	*5*
1	100	140	120	80	100	0	0	-160	0	-1634
2	-	140	280	240	260	-	0	0	160	-1474
3	-	-	360	160	260	-	-	80	80	-1474
4	-	-	-	80	580	-	-	-	0	-1154
5	-	-	-	-	1734	-	-	-	-	0

TABLE 3

Details of the Enumeration Tree

Vertex v	*Corresponding Node in* $\mathcal{G}^G$	θ_v	δ_v	ϕ_v	z_v	*Remarks*
0	T					
1	V_{45}	0.275	0.001250	850		
2	V_{55}	0.025	0.000562	800		Incumbent, $UB = 800$
3	V_{44}	0.025	0.000562	800	0.000	
4	V_{13}	0.475	0.001250	900	-0.200	Fathomed, $\phi_v \geq UB + F_2$
5	V_{23}	0.275	0.001250	850	0.000	
6	V_{33}	0.175	0.001250	900	0.100	Fathomed, $\phi_v \geq UB + F_2$
7	V_{13}	0.115	0.000562	850	-0.090	
8	V_{23}	0.025	0.000562	800	0.000	
9	V_{33}	0.980	0.000562	750	0.045	Current incumbent Revised $UB = 750$
10	V_{11}	0.275	0.001250	850	0.000	
11	V_{11}	0.025	0.000562	800	0.000	
12	V_{12}	0.980	0.000562	800	0.000	
13	V_{22}	0.980	0.000562	750	0.000	
14	V_{22}	0.980	0.000562	750	0.000	

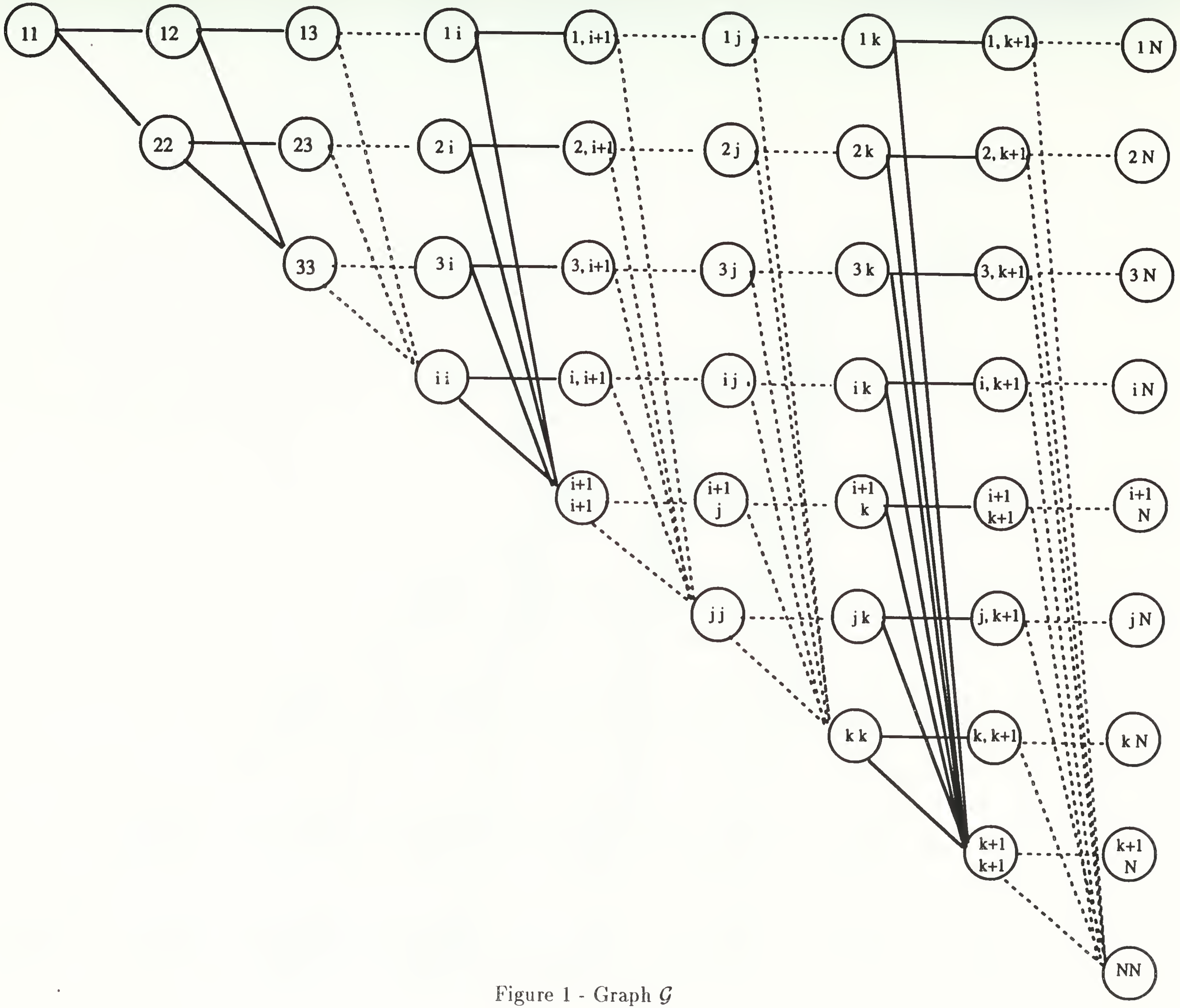

Figure 1 - Graph $\mathcal{G}$

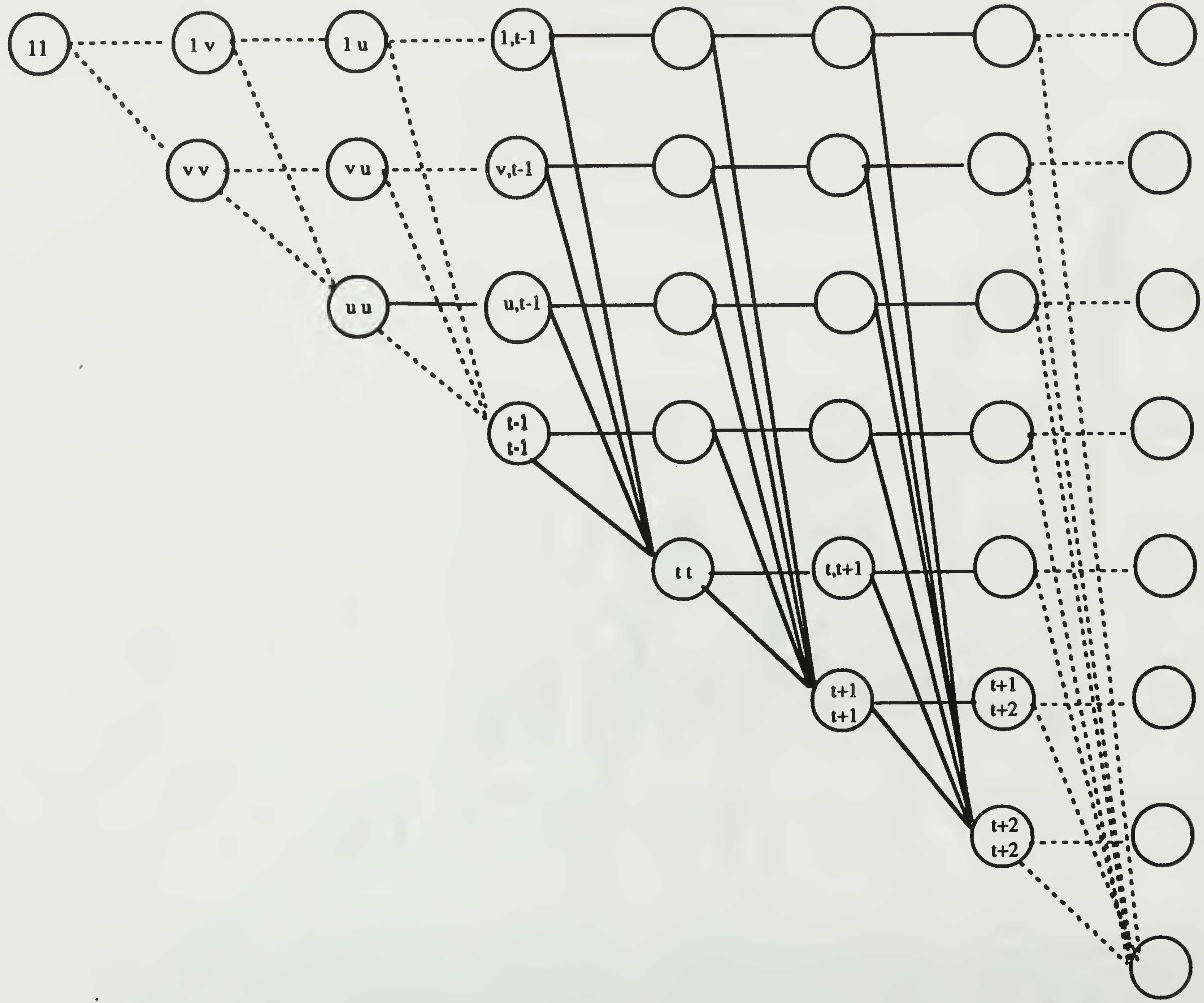

Figure 2 - Proof of Lemma 3

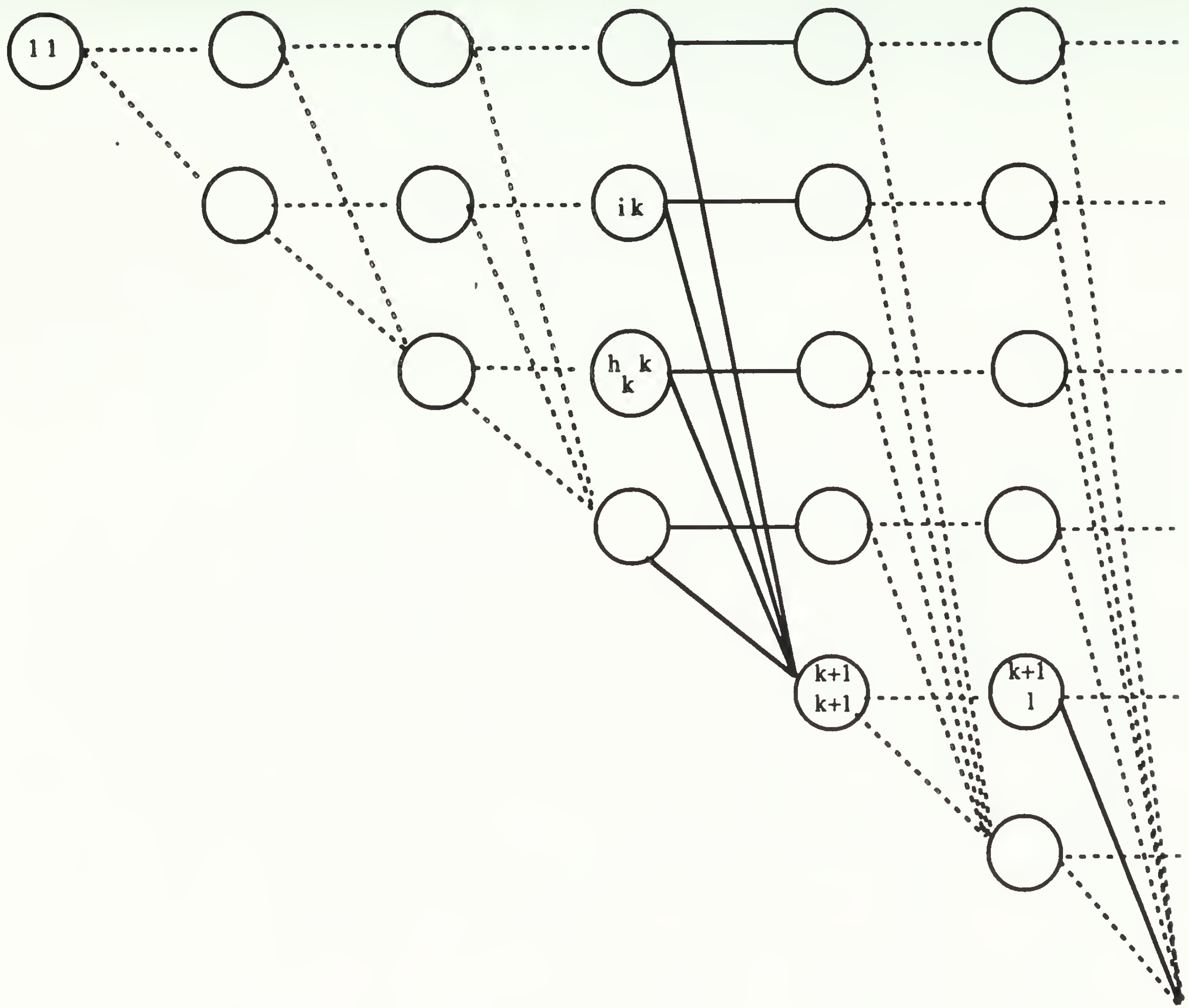

Figure 3a - Proof of Lemma 4

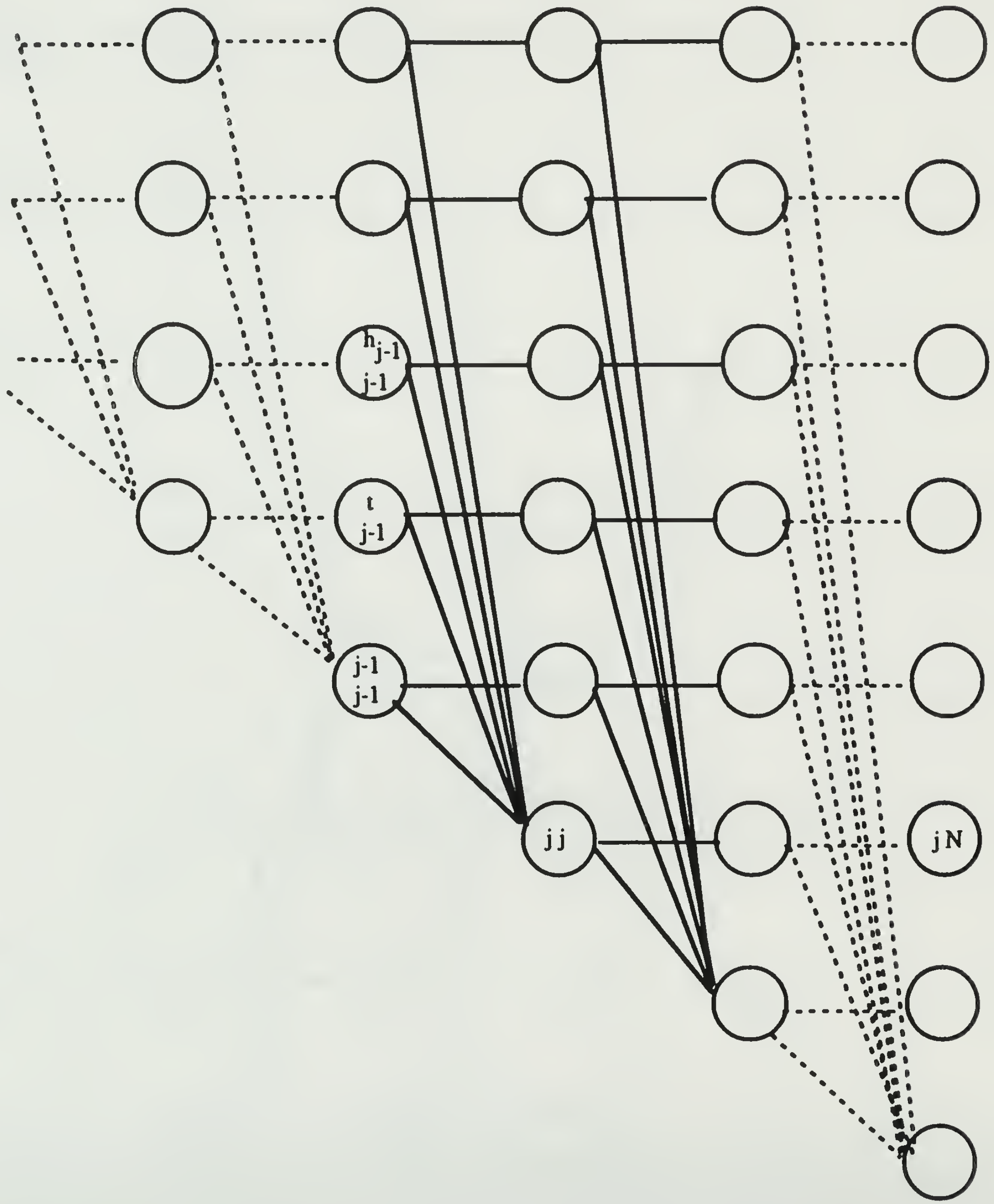

Figure 3b - Proof of Lemma 4 (continued)

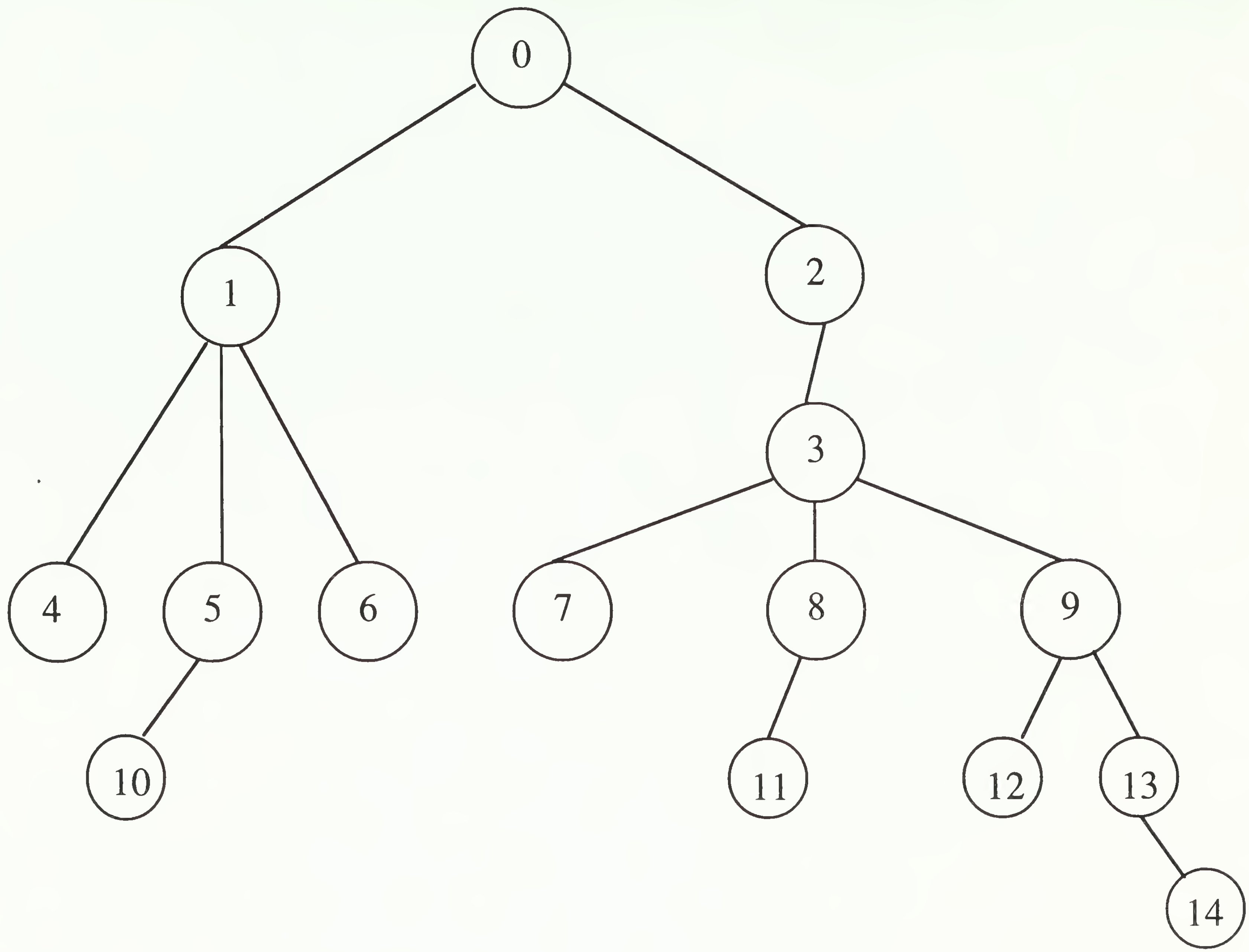

Figure 4 - Enumeration Tree for the Example Problem

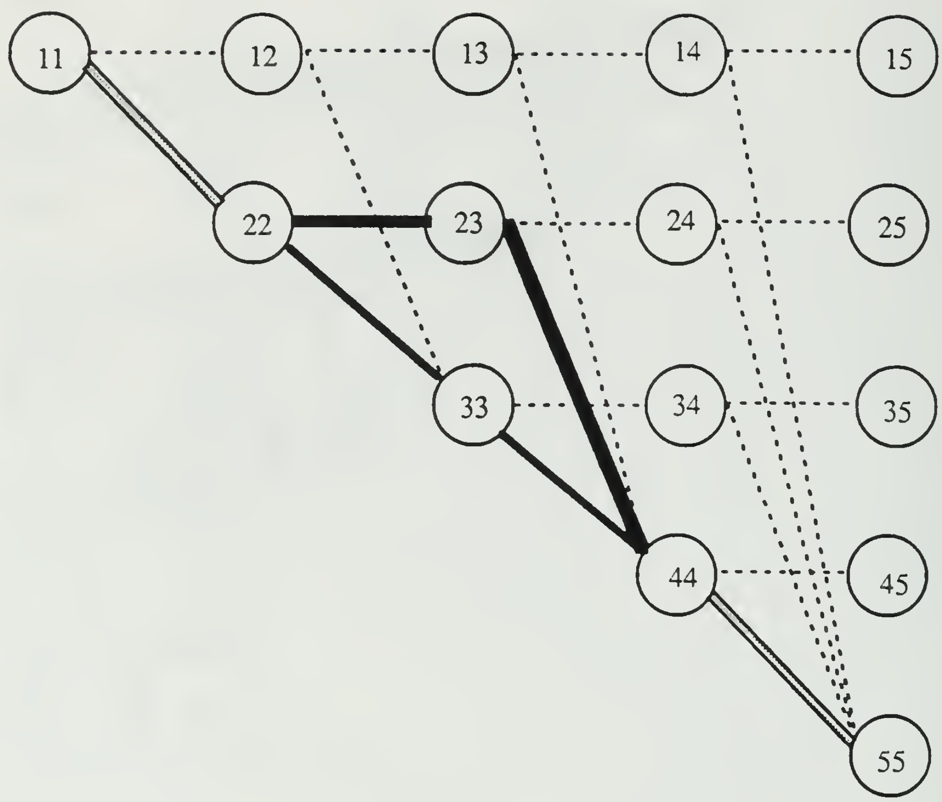

Figure 5 - Paths Ω^G and Ω^* in the Example Problem