V-SHAPED POLICIES FOR SCHEDULING DETERIORATING JOBS

GUR MOSHEIOV

Hebrew University, Jerusalem, Israel and Baruch College of the City University of New York

(Received July 1988; revisions received February, December 1989, April 1990; accepted April 1990)

A set of N jobs has to be processed on a single machine. Jobs have the same basic processing time, but the actual processing time of each job grows linearly with its starting time. A (possibly) different rate of growth is associated with each job. We show that the optimal sequence to minimize flow time is V-shaped: Jobs are arranged in descending order of growth rate if they are placed before the minimal growth rate job, and in ascending order if placed after it. Efficient ($0(N \log N)$) asymptotically optimal heuristics are developed. Their average performance is shown to be extremely good: The average relative error over a set of 20-job problems is on the order of 10^{-5}.

Browne and Yechiali (1990) introduced a scheduling problem in which the processing times of the jobs are not constant over time. N jobs have to be processed on a single machine to minimize the makespan. Job i is characterized by: i) a "basic" processing time p_i, the length of time required to complete the job if it were scheduled first, i.e., at $t = 0$, and ii) a parameter α_i that, jointly with p_i, determines the job's (actual) processing time at $t > 0$. α_i can be interpreted as the growth rate of the processing time of job i. Assuming linear deterioration, i.e., the processing time of the job increases linearly with its starting time t, the actual processing time is $p_i + \alpha_i t$. Browne and Yechiali found that the optimal index policy to minimize the makespan is to schedule jobs in an increasing order of p_i/α_i, the ratio of the basic processing time to the growth rate.

In this context, it is natural to investigate the properties of optimal policies under alternative measures of performance. The objective function studied in this paper is of minimizing flow time. We deal with a special case: basic processing times are assumed to be equal for all jobs ($p_i = p$ for all i). We show that the optimal policy for this case has a unique form which reduces significantly the computational effort of the optimization process. For large-sized problems (in which many jobs have to be scheduled), extremely good heuristics are developed.

The motivation for analyzing this case arises not only from the intrinsic interest in it per se, but also because it serves as a good approximation to the general case (distinct p_i's) when the number of jobs is large. This follows from the fact that as N increases, the starting time of many jobs is large and p_i becomes negligible. (For large t $p_i + \alpha_i t \sim \alpha_i t$. Therefore, when $N \to \infty$ and all processing times are positive, the actual processing time of infinitely many jobs is not affected by the basic processing time.)

Deterioration in processing time may occur when the machine is losing efficiency while processing a batch of jobs. At $t = 0$ the machine is assumed to be at maximal efficiency. The efficiency loss is reflected in the fact that a job which is processed later in time has a longer processing time.

Other applications are in the area of scheduling maintenance or cleaning assignments. In many real-life situations, a delay in maintenance or cleaning implies an additional effort (= time) to accomplish it. Fixed increasing rates of this effort, i.e., "linear deterioration," is a simplifying but reasonable assumption. Clearly, the increasing rate of different jobs (e.g., at different locations) may be significantly different.

The main result of this study is that the optimal policy has a V-shape: Jobs are arranged in descending order of growth rates if they are placed before the minimal growth rate job, but in ascending order if placed after it. This property enables us to get exact solutions for problems of relatively large size. In addition, simple and efficient heuristics ($0(N \log N)$) are developed. Our computational experiments show that these heuristics provide outstanding results. The average relative error in our set of experiments (20-job problems) was of the order 10^{-5}. As the number of jobs increases, this error decreases to zero (asymptotic optimality).

Two generalizations have also been studied: scheduling deteriorating jobs with unequal basic processing times ($p_i \neq p_j$ for $i \neq j$), and scheduling deteriorating jobs with unequal weights (to minimize the sum of

Subject classification Production/scheduling: deterministic sequencing, single machine.

0030-364X/91/3906-0979 $01.25
© 1991 Operations Research Society of America

Copyright © 2001 All Rights Reserved

weighted completion times). The V-shaped property does not generalize to any of these cases.

V-shaped optimal policies are quite rare in the scheduling literature. Eilon and Chowdhury (1977) were the first to prove that a V-shaped policy is optimal (for the waiting time variance minimization problem). They verified an earlier conjecture of Merten and Muller (1972) and Schrage (1975). In the last decade, several other problems were shown to be optimized by a V-shaped sequence. Most of these fall in the category of minimization of the sum of deviations about a common due date. Kanet (1981), Sundaraghavan and Ahmed (1984), Bagchi, Sullivan and Chang (1986), and Hall (1987) studied the minimization of absolute deviations about a common due date. Panwalker, Smith and Seidmann (1982), Emmons (1987), and Bagchi, Chang and Sullivan (1987) studied the asymmetric version (different earliness and tardiness penalties) of this problem. Bagchi, Sullivan and Chang (1987) showed an optimal V-shaped sequence for the minimization of the sum of squared deviations about a common due date, an equivalent problem to variance minimization. Hall and Posner (1991) studied the problem of weighted earliness and tardiness (job dependent penalties). "Restriction versions" of some of these problems (i.e., when the due date is early enough to affect the scheduling decision), were also studied, and the V-shaped property was shown to preserve. For a complete list of studies see the survey of Baker and Scudder (1990). In all these examples, the V-shaped property (the LPT–SPT structure is a necessary condition for optimality. An example of a different type is the SEPT–LEPT policy (shortest expected processing time, longest expected processing time), to minimize the expected makespan in stochastic flow-shops (Pinedo 1982). In contrast to the former cases, here V-shape is also sufficient for optimality (all SEPT–LEPT policies result in the same expected makespan and all are optimal).

The paper is organized as follows. In Section 1 we provide the formulation of the problem. The V-shape and some other properties of the optimal policy are proven in Section 2. Section 3 examines the impact of these properties on the reduction of the computational effort. Two examples are given. Heuristics are developed in Section 4 and the results of their empirical evaluation are presented in Section 5.

1. FORMULATION

Our basic assumption is that all basic processing times are equal. Without affecting the optimal policy we may assume that they are all equal to one unit of time: $p_i = 1$, $i = 1, \ldots, N$. Jobs are, therefore, characterized only by their growth rates α_i. The actual processing time of job i is $1 + \alpha_i t$, where t is the current time.

The following formulation was introduced (for the makespan case) by Browne and Yechiali. Let Π denote the set of all $N!$ permutations of the set $\{1, 2, \ldots, N\}$; $\pi \in \Pi$ denotes an arbitrary permutation. Denote by π_0 the permutation $(1, 2, \ldots, N)$. Let Y_i be the actual processing time of the ith job of the sequence π_0.

Then

$$Y_1 = 1$$

and

$$Y_i = 1 + \alpha_i \sum_{j=1}^{i-1} Y_j \quad i = 2, \ldots, N.$$

The completion time of the ith job is:

$$S_i = \sum_{k=1}^{i} Y_k = \sum_{k=1}^{i} \left(1 + \alpha_k \sum_{j=1}^{k-1} Y_j\right) \quad i = 1, \ldots, N.$$

It is easy to verify that

$$\sum_{k=1}^{i} \left(1 + \alpha_k \sum_{j=1}^{k-1} Y_j\right) = \sum_{k=1}^{i} \prod_{j=k+1}^{i} (1 + \alpha_j) \qquad i = 1, \ldots, N.$$

Therefore, the total flow-time is given by:

$$F(\pi_0) = F = \sum_{i=1}^{N} S_i = \sum_{i=1}^{N} \sum_{k=1}^{i} \prod_{j=k+1}^{i} (1 + \alpha_j).$$

2. PROPERTIES OF THE OPTIMAL SEQUENCE

A trivial property of the optimal sequence is as follows.

Proposition 1 (scheduling the first job). *Let $k = \arg\max\{\alpha_i, i = 1, \ldots, N\}$. Then k is the first job in the optimal sequence.*

Proof. The processing time of the first scheduled job is $1 + \alpha_{(1)} t = 1$ $(t = 0)$. $\alpha_{(1)}$ does not affect the flow time and therefore, independently of the rest of the sequence, it is always better to first schedule the job with the largest growth rate.

Given this result, we are left with the problem of scheduling the remaining $N - 1$ jobs. For ease of exposition define the set of jobs $A = S \backslash \{k\}$, where S is the original complete set and k is defined

Copyright © 2001 All Rights Reserved

in Proposition 1. Denote $N - 1$ by M and renumber the jobs of set A to have indices from 1 to M. Denote $1 + \alpha_j$ by a_j, $j = 1, \ldots, M$. Then the problem is stated as follows.

Given the set $A = \{a_1, a_2, \ldots, a_M\}$, $a_j \geq 1$, $j = 1, \ldots, M$, find the sequence $(a_{(1)}, a_{(2)}, \ldots, a_{(M)})$ to minimize

$$g(\pi) = \sum_{i=1}^{M} \sum_{k=1}^{i} \prod_{j=k}^{i} a_{(j)}.$$

Note that for a given sequence, the original flow time F is larger than the value of g by the constant N. That follows because the first job's processing time is always 1 and it is included in the completion times of all N jobs. The function g refers only to the set A and does not include this fixed quantity.

The remainder of this section deals with scheduling the jobs of the set A and the properties of the function g.

Proposition 2 (symmetry). *For any sequence π let $\bar{\pi}$ be the reversed sequence. Then $g(\pi) = g(\bar{\pi})$.*

Proof. For convenience let $\pi = \pi_0 = (1, 2, \ldots, M)$.

$$\begin{aligned} g(\tau_0) = a_1 & \\ &+ a_1 a_2 + a_2 \\ &+ a_1 a_2 a_3 + a_2 a_3 + a_3 \\ &\vdots \\ &+ a_1 a_2 a_3 \ldots a_{M-1} + \ldots + a_{M-1} \\ &+ a_1 a_2 a_3 \ldots a_{M-1} a_M + \ldots + a_{M-1} a_M + a_M \end{aligned}$$

$$\begin{aligned} g(\bar{\pi}_0) = a_M & \\ &+ a_M a_{M-1} + a_{M-1} \\ &\vdots \\ &+ a_M a_{M-1} \ldots a_2 + \ldots + a_2 \\ &+ a_M a_{M-1} \ldots a_2 a_1 + \ldots + a_2 a_1 + a_1 \end{aligned}$$

and the above expressions are equal.

An immediate implication of Proposition 2 is that the optimal solution is not unique. From the symmetry property of the function g (which is defined on the set A), if the sequence $(1, 2, \ldots, N)$ is optimal, then so is the sequence $(1, N, N - 1, \ldots, 2)$.

Lemma 1. *Let $l = \arg\min_{i \in A} \{a_i\}$. Then, within the set A, l is scheduled neither first nor last in the optimal sequence.*

Proof. Consider any sequence with job l placed first. For convenience let

$\pi_1 = (l, 2, 3, \ldots, M)$.

π_2 is the schedule obtained by interchanging the first two jobs:

$\pi_2 = (2, l, 3, \ldots, M)$.

Then,

$$g(\pi_2) - g(\pi_1) = (a_l - a_2) \sum_{k=3}^{M} \prod_{j=3}^{k} a_j.$$

Since $a_l \leq a_2$ the above expression is nonpositive and therefore π_2 is a better policy. From the symmetry of the function g (Proposition 1), job l cannot be scheduled last as well.

Recall that we are dealing with scheduling jobs within the set A. Thus, job l can be scheduled neither *second* nor last in the optimal (complete) sequence.

Lemma 2. *Let a_{i-1}, a_i, a_{i+1} be three consecutive numbers in a sequence. If $a_i > a_{i-1}$ and $a_i > a_{i+1}$ the sequence is not optimal.*

Proof. We show that an interchange between a_{i-1} and a_i or between a_i and a_{i+1} reduces the value of g. Let

$\pi_0 = (1, 2, \ldots, i - 1, i, i + 1, \ldots, M)$

$\pi_1 = (1, 2, \ldots, i - 2, i, i - 1, i + 1, \ldots, M)$

$\pi_2 = (1, 2, \ldots, i - 1, i + 1, i, i + 2, \ldots, M)$

$$g(\pi_1) - g(\pi_0) = (a_i - a_{i-1}) \sum_{k=1}^{i-2} \prod_{j=k}^{i-2} a_j + (a_{i-1} - a_i) \sum_{k=i+1}^{M} \prod_{j=i+1}^{k} a_j \tag{1}$$

$$g(\pi_2) - g(\pi_0) = (a_{i+1} - a_i) \sum_{k=1}^{i-1} \prod_{j=k}^{i-1} a_j + (a_i - a_{i+1}) \sum_{k=i+2}^{M} \prod_{j=i+2}^{k} a_j. \tag{2}$$

We show that (1) and (2) cannot both be positive. Let

$$X = \sum_{k=1}^{i-2} \prod_{j=k}^{i-2} a_j$$

$$Y = \sum_{k=i+2}^{M} \prod_{j=i+2}^{k} a_j.$$

Copyright © 2001 All Rights Reserved

Then both terms are simplified:

$$(1) = (a_i - a_{i-1})X + (a_{i-1} - a_i)a_{i+1}(Y + 1).$$

$$(2) = a_{i-1}(X + 1)(a_{i+1} - a_i) + (a_i - a_{i+1})Y.$$

If both expressions are positive then,

$$X > a_{i+1}(Y + 1),$$

$$Y > a_{i-1}(X + 1),$$

which is a contradiction since $a_i \geq 1$, $i = 1, \ldots, M$. Therefore either π_1 or π_2 are better policies than π_0.

Theorem 1. (V-shape). *The optimal sequence has a* V-*shape, i.e., jobs are arranged in descending order if they are placed before the job with the smallest* a_i *but in ascending order if placed after it.*

Proof. It is straightforward from the previous two lemmas.

To show the next property we need the following definition: a sequence is *perfectly symmetric* V-*shaped* if it is V-shaped and $a_i = a_{M-i+1}$, $i = 1, \ldots, M$. In terms of the original set of growth rates $\{\alpha_1, \alpha_2, \ldots, \alpha_N\}$, perfectly symmetric V-shaped means $\alpha_i = \alpha_{N-i+2}$, $i = 2, \ldots, N$. Note that the first scheduled job does not have a corresponding job with equal growth rate. For example, given the set $\{\alpha_0, \alpha_1, \alpha_1, \alpha_2, \alpha_2, \alpha_3, \alpha_3\}$, where $\alpha_0 \geq \alpha_1 \geq \alpha_2 \geq \alpha_3$, the sequence (0, 1, 2, 3, 3, 2, 1) is perfectly symmetric V-shaped. Given the set $\{\alpha_0, \alpha_1, \alpha_1, \alpha_2, \alpha_2, \alpha_3\}$, the sequence (0, 1, 2, 3, 2, 1) is perfectly symmetric V-shaped. On the other hand, given the set $\{\alpha_1, \alpha_1, \alpha_2, \alpha_2, \alpha_3, \alpha_3\}$, the sequence (1, 2, 3, 3, 2, 1) is *not* perfectly symmetric according to our definition.

Proposition 3. (perfectly symmetric V-shape). *If a perfectly symmetric* V-*shaped sequence can be constructed from the set of jobs, then it is optimal.*

Proof. The proof is given in the Appendix.

To summarize, the optimal sequence has the following properties:

1. it has a V-shape;
2. the first job has the largest value of α;
3. the flow time function is symmetric for the set of jobs placed second through last.

We also show that a perfectly symmetric V-shaped sequence is always optimal.

3. COMPLEXITY REDUCTION

The V-shaped property of the optimal sequence reduces significantly the computational effort needed to obtain an exact solution to the problem. The number of V-shaped sequences of length N is equivalent to the total number of subsets of the set $\{1, 2, \ldots, N\}$. Any subset can be transformed into a V-shaped sequence (and vice versa) by arranging the numbers of the subset in descending order and adding the numbers of the complement subset in ascending order. Therefore, instead of checking all $N!$ permutations, only an effort of $0(2^N)$ is needed.

In our case, even further reductions are possible. Recall that the job with the largest growth rate is scheduled first. The V-shape and the flow time symmetry of the remaining $N - 1$ job set implies that there exists an optimal sequence in which the job with the second largest α is scheduled last. (Another optimal sequence contains this job in the second place.) Assuming that it is placed last (as we did in our experiments), the job with the third largest α must be scheduled second or next to last. The V-shape of the remaining $N - 1$ job set also implies that the job with the lowest value of α cannot be scheduled second. It is easy to verify that the total number of sequences that are candidates for optimality is, in fact, $2^{N-3} - 1$.

Our experiments show that even this magnitude does not reflect the real effort: As the following two examples indicate, some sequences having all the above properties are never optimal.

Example 1: A 5-Job Sequence

For convenience, assume that $\alpha_1 \geq \alpha_2 \geq \alpha_3 \geq \alpha_4 \geq \alpha_5$. By Proposition 1, job 1 is scheduled first. By Proposition 2 and Theorem 1 there is an optimal sequence in which job 2 is scheduled last. Job 3 must be scheduled second or next to last. By Theorem 1, job 5 cannot be scheduled second. Therefore, the V-shaped relevant sequences ($2^{5-3} - 1 = 3$) are:

$$\pi_1 = (1, 3, 4, 5, 2)$$

$$\pi_2 = (1, 3, 5, 4, 2)$$

$$\pi_3 = (1, 4, 5, 3, 2)$$

$$\begin{aligned} F(\pi_1) - F(\pi_2) &= (1 + \alpha_3)(1 + \alpha_4) + (1 + \alpha_5)(1 + \alpha_2) \\ &\quad - (1 + \alpha_3)(1 + \alpha_5) - (1 + \alpha_4)(1 + \alpha_2) \\ &= (\alpha_3 - \alpha_2)(\alpha_4 - \alpha_5) \leq 0 \end{aligned}$$

Copyright © 2001 All Rights Reserved

$$F(\pi_2) - F(\pi_3)$$
$$= (1 + \alpha_4)(1 + \alpha_2) + (1 + \alpha_5)(1 + \alpha_4)$$
$$- (1 + \alpha_3)(1 + \alpha_2) - (1 + \alpha_5)(1 + \alpha_3)(1 + \alpha_2)$$
$$= (\alpha_4 - \alpha_3)[1 + \alpha_2 + (1 + \alpha_5)(1 + \alpha_2)] \leq 0.$$

Therefore, policy $\pi_1 = (1, 3, 4, 5, 2)$ is optimal for any 5-job problem.

Example 2: A 6-Job Sequence

Assume that $\alpha_1 \geq \alpha_2 \geq \alpha_3 \geq \alpha_4 \geq \alpha_5 \geq \alpha_6$. The number of possible optimal sequences is $2^{6-3} - 1 = 7$. The candidates are:

$$\pi_1 = (1, 3, 4, 5, 6, 2)$$
$$\pi_2 = (1, 3, 4, 6, 5, 2)$$
$$\pi_3 = (1, 3, 5, 6, 4, 2)$$
$$\pi_4 = (1, 4, 5, 6, 3, 2)$$
$$\pi_5 = (1, 5, 6, 4, 3, 2)$$
$$\pi_6 = (1, 4, 6, 5, 3, 2)$$
$$\pi_7 = (1, 3, 6, 5, 4, 2)$$

$$F(\pi_1) - F(\pi_7)$$
$$= (1 + \alpha_3)(1 + \alpha_4) + (1 + \alpha_6)(1 + \alpha_2)$$
$$+ (1 + \alpha_3)(1 + \alpha_4)(1 + \alpha_5)$$
$$+ (1 + \alpha_5)(1 + \alpha_6)(1 + \alpha_2)$$
$$- [(1 + \alpha_3)(1 + \alpha_6) + (1 + \alpha_4)(1 + \alpha_2)$$
$$+ (1 + \alpha_3)(1 + \alpha_6)(1 + \alpha_5)$$
$$+ (1 + \alpha_5)(1 + \alpha_4)(1 + \alpha_2)]$$
$$= (\alpha_4 - \alpha_6)(\alpha_3 - \alpha_2)(2 + \alpha_5) \leq 0.$$

In a similar way we can show that:

$$F(\pi_6) - F(\pi_5) \leq 0$$
$$F(\pi_4) - F(\pi_6) \leq 0$$
$$F(\pi_2) - F(\pi_3) \leq 0$$
$$F(\pi_2) - F(\pi_4) \leq 0$$

Therefore, for a 6-job problem either $\pi_1 = (1, 3, 4, 5, 6, 2)$ or $\pi_2 = (1, 3, 4, 6, 5, 2)$ is optimal.

4. HEURISTICS

Although major reductions can be achieved as described above, the V-shaped property does not reduce the computational effort to be less than exponential with the number of jobs. Heuristics are the main tool for solving problems containing a large number of jobs.

Two heuristics are proposed. In the first, we create a V-shaped sequence with the same number of jobs in both sides of the minimum. The idea of the second heuristic is to get a V-shaped sequence with an (approximately) equal sum of growth rates in both sides of the minimum.

Define "section 1" to be the set of jobs placed before the job with minimal growth rate and "section 2" to be the set of jobs placed after it.

Heuristic 1

Step 1. Arrange the jobs in descending order of growth rates. Call this the descending sequence (DS).

Step 2. Assign the first job of DS to be scheduled first, the second job of DS to be scheduled last, the third job of DS to be scheduled second, and the fourth job of DS to be scheduled next to last. Continue by adding jobs alternately to each section (add to the first available place in section 1 and to the last available place in section 2).

Heuristic 2

Step 1. Arrange the jobs in descending order of growth rates (DS).

Step 2. Assign the first job of DS to be scheduled first, the second job of DS to be scheduled last, and the third job of DS to be scheduled next to last. Let SUM1 be the current sum of growth rates in section 1 and SUM2 to be the current sum of growth rates in section 2. Continue by adding at each step the next job of DS to section 1 (in the first available place) if SUM1 $\leq$ SUM2 and to section 2 (in the last available place) if SUM2 $<$ SUM1.

Running Time. Both heuristics are efficient ($0(N \log N)$) and easy to implement.

Asymptotic Optimality. Both heuristics are asymptotically optimal under the assumption that all α_i's are i.i.d. random variables. Denote by π^* the optimal sequence and by π_1 and π_2 the sequences created by heuristics 1 and 2, respectively. Let $\Pi_v \subseteq \pi_v$ be the set of all V-shaped sequences. For any $\pi_v \in \Pi_v$ we define a "measure of closeness to a perfectly symmetric V":

$$c(\pi) = \max_{1 \leq k \leq n} |\alpha_{(k)} - \alpha_{(N-k+1)}|.$$

For a given set of jobs S we define $c^*(s) = \min_{\pi \in \Pi_v} c(\pi)$, i.e., the minimal value of c that can be achieved

Copyright © 2001 All Rights Reserved

by any V-shaped sequence π constructed from the set S.

First we claim that $\lim_{N\to\infty} c^*(s) = 0$ a.s. (As N increases it is always possible to find a "more symmetric V-shaped" sequence.) In order to show this, let $S^1 = \{\alpha_1^1, \alpha_2^1, \ldots, \alpha_{N/2}^1\}$ and $S^2 = \{\alpha_1^2, \alpha_2^2, \ldots, \alpha_{N/2}^2\}$ be two sets of i.i.d. random variables (assume that N is even) and let $\alpha_{(j)}^1$ and $\alpha_{(j)}^2$ be the jth order statistics of S^1 and S^2, respectively. In any realistic case α has a distribution of finite support. Thus, it is clear that $\lim_{N\to\infty} |\alpha_{(j)}^1 - \alpha_{(j)}^2| = 0$ a.s., $1 \leq j \leq N/2$. Let $S = S^1 \cup S^2$ and construct the V-shaped sequence

$$\pi = (\alpha_{(N/2)}^1, \alpha_{[(N/2)-1]}^1, \ldots, \alpha_{(1)}^1, \alpha_{(1)}^2, \alpha_{(2)}^2, \ldots, \alpha_{(N/2)}^2).$$

It follows that $c^*(s) \to 0$ a.s. as $N \to \infty$.

Next we claim:

$$\lim_{N\to\infty} c(\pi_1) = 0 \text{ a.s.}, \quad \lim_{N\to\infty} c(\pi_2) = 0 \text{ a.s.}$$

In the above we showed, in fact, that as $N \to \infty$ one can divide the set S into two sets S^1 and S^2 of size $N/2$ each, such that $|\alpha_{(j)}^1 - \alpha_{(j)}^2| \to 0$ a.s.; $\alpha_{(j)}^1 \in S^1$, $\alpha_{(j)}^2 \in S^2$, $j = 1, \ldots, N/2$. Note that $\alpha_{(j)}^1$ and $\alpha_{(j)}^2$ appear as consecutive numbers in the sorted set obtained after Step 1 of both heuristics. By Step 2 of Heuristic 1 these jobs will indeed be "assigned" to different sections. Assigning them to different sections also minimizes the difference between the "current sums of growth rates" of the sections. Thus (asymptotically) Heuristic 2 results in the same policy. Therefore as $N \to \infty$, $c(\pi_1) \to 0$ and $c(\pi_2) \to 0$ a.s.

Proposition 3 states that if a perfectly symmetric V-shaped sequence can be achieved, it is optimal, i.e., if $c(\pi) = 0$ for some $\pi \in \Pi_v$, then $\pi = \pi^*$. Asymptotic optimality is a straightforward result of the above two claims and Proposition 3.

5. COMPUTATIONAL EXPERIMENTS

Four sets of problems are examined: 10-job problems (Table I), 20-job problems (Table II), 100-job problems (Table III) and 250-job problems (Table IV). In the first two sets the optimal and the heuristic solutions are obtained. In the last two sets only the heuristic solutions are presented and the purpose is to examine their "asymptotic" behavior. Each set consists of 25 problems. All growth rates are randomly generated from a uniform distribution ($\alpha_i \sim U(0, 1)$).

Denote by F^* the optimal flow time and by H_1 and

Table I
10-Job Problems

Problem No.	F^*	H_1	H_2	H_1/F^*	H_2/F^*	H_1/H_2
1	268.587	268.760	268.587	1.000644	1.000000	1.000644
2	313.481	313.942	313.487	1.001471	1.000019	1.001451
3	378.138	378.807	378.300	1.001769	1.000428	1.001340
4	161.826	162.480	161.940	1.004041	1.000704	1.003335
5	186.287	186.569	186.287	1.001514	1.000000	1.001514
6	191.941	192.198	191.941	1.001339	1.000000	1.001339
7	226.458	226.671	226.458	1.000941	1.000000	1.000941
8	160.131	160.568	160.706	1.002729	1.003591	0.999141
9	134.483	134.809	134.724	1.002424	1.001792	1.000631
10	136.806	136.907	136.806	1.000738	1.000000	1.000738
11	490.284	490.817	490.373	1.001087	1.000182	1.000905
12	148.260	148.825	148.366	1.003811	1.000715	1.003094
13	234.264	234.493	234.264	1.000978	1.000000	1.000978
14	184.670	184.830	184.670	1.000866	1.000000	1.000866
15	292.814	293.180	292.814	1.001250	1.000000	1.001250
16	142.815	142.932	142.819	1.000819	1.000028	1.000791
17	183.042	183.129	183.042	1.000475	1.000000	1.000475
18	234.867	235.171	234.867	1.001294	1.000000	1.001294
19	160.595	160.940	160.595	1.002148	1.000000	1.002148
20	134.993	135.262	135.014	1.001993	1.000156	1.001837
21	325.623	326.291	325.623	1.002051	1.000000	1.002051
22	277.566	277.896	277.566	1.001189	1.000000	1.001189
23	142.049	142.710	142.087	1.004653	1.000268	1.004385
24	195.601	195.651	195.601	1.000256	1.000000	1.000256
25	189.002	189.203	189.002	1.001063	1.000000	1.001063

Copyright © 2001 All Rights Reserved

Table II
20-Job Problems

Problem No.	F^*	H_1	H_2	H_1/F^*	H_2/F^*	H_1/H_2
1	8493.8359	8506.3555	8493.9883	1.001474	1.000018	1.001456
2	12418.3359	12435.8281	12418.7031	1.001409	1.000030	1.001379
3	7182.7305	7197.2188	7182.9297	1.002017	1.000028	1.001989
4	5733.3203	5736.9336	5733.5352	1.000630	1.000037	1.000593
5	5689.8555	5692.4141	5689.9375	1.000450	1.000014	1.000435
6	10242.9766	10250.4727	10243.0352	1.000732	1.000006	1.000726
7	4070.6589	4081.6199	4071.4775	1.002693	1.000201	1.002491
8	4074.2898	4090.9885	4074.3806	1.004099	1.000022	1.004076
9	2888.0208	2893.7456	2888.0315	1.001982	1.000004	1.001979
10	52976.3750	53002.9840	52976.5469	1.000502	1.000003	1.000499
11	4840.1914	4847.1875	4840.5820	1.001445	1.000081	1.001365
12	8573.3750	8584.4141	8573.3750	1.001288	1.000000	1.001288
13	6823.7383	6832.9023	6824.1133	1.001343	1.000055	1.001288
14	13981.1602	13999.8633	13981.2461	1.001338	1.000006	1.001332
15	7552.9688	7563.1445	7553.1758	1.001347	1.000027	1.001320
16	7871.9609	7882.0742	7872.2227	1.001285	1.000033	1.001251
17	6564.5781	6573.8672	6565.0000	1.001415	1.000064	1.001351
18	5396.6914	5411.7734	5396.7266	1.002795	1.000007	1.002788
19	5029.6406	5033.6055	5029.6406	1.000788	1.000000	1.000788
20	20945.2930	20953.5117	20945.6484	1.000392	1.000017	1.000375
21	7858.6523	7874.2930	7859.4766	1.001990	1.000105	1.001885
22	3916.0310	3919.1714	3916.1714	1.000802	1.000036	1.000766
23	14939.1367	14957.8008	14940.7656	1.001249	1.000109	1.001140
24	10252.4336	10268.3086	10252.4492	1.001548	1.000002	1.001547
25	5825.1445	5834.4648	5825.2266	1.001600	1.000014	1.001586

H_2 the results of Heuristic 1 and Heuristic 2, respectively. We use the measures H_1/F^* and H_2/F^* to evaluate the performance of the heuristics and H_1/H_2 to compare them.

(All runs were done on IBM 4341. The average CPU time needed to get the exact solution of a 20-job problem was 219 seconds. The average CPU time needed to heuristically solve a 250-job problem was 0.3 second.)

Our basic goal in these experiments is to test the average performance of the heuristics by measuring the relative error on different sets of problems. Other issues of interest are: a comparison between the heuristics, and their degree of accuracy as the number of jobs increases. Table V summarizes the *average values* of the performance measures in all our experiments.

An immediate conclusion from Table V (10-job and 20-job sets) is that both heuristics perform outstandingly. The average relative error of the first heuristic is less than 0.2% in both sets of problems. The average relative error of the second heuristic is less than 0.04% in the 10-job problems and less than 0.004% (an order of 10^{-5}) in the 20-job problems. Recall that the effort needed to reach these results is only $0(N \log N)$.

Table III
100-Job Problems

Problem No.	H_1	H_2	H_1/H_2
1	1.744097E + 18	1.744085E + 18	1.0000069
2	1.053412E + 17	1.053396E + 17	1.0000150
3	4.843994E + 17	4.843964E + 17	1.0000061
4	1.434196E + 17	1.434169E + 17	1.0000187
5	5.246065E + 17	5.246037E + 17	1.0000054
6	4.363240E + 17	4.363174E + 17	1.0000151
7	5.628693E + 16	5.628650E + 16	1.0000076
8	1.575655E + 16	1.575634E + 16	1.0000136
9	2.509319E + 16	2.509291E + 16	1.0000113
10	2.628011E + 17	2.627961E + 17	1.0000191
11	3.076352E + 17	3.076289E + 17	1.0000206
12	2.806704E + 16	2.806607E + 16	1.0000349
13	1.420805E + 17	1.420790E + 17	1.0000106
14	5.717080E + 17	5.716959E + 17	1.0000213
15	1.774278E + 18	1.774267E + 18	1.0000062
16	4.399426E + 17	4.399398E + 17	1.0000064
17	5.463203E + 16	5.463132E + 16	1.0000130
18	7.612194E + 16	7.612084E + 16	1.0000144
19	3.197830E + 18	3.197803E + 18	1.0000083
20	5.350885E + 17	5.350851E + 17	1.0000064
21	4.052247E + 17	4.052187E + 17	1.0000149
22	1.638351E + 16	1.638286E + 16	1.0000393
23	3.558640E + 17	3.558580E + 17	1.0000170
24	8.976674E + 16	8.976420E + 16	1.0000283
25	8.086475E + 17	8.086382E + 17	1.0000116

Copyright © 2001 All Rights Reserved

Table IV
250-Job Problems

Problem No.	H_1	H_2	H_1/H_2
1	3.66738922E + 42	3.66737858E + 42	1.0000029
2	2.35548133E + 43	2.35546857E + 43	1.0000054
3	3.54127042E + 39	3.54127886E + 39	0.9999976
4	5.74418833E + 42	5.74417637E + 42	1.0000021
5	2.94696652E + 43	2.94694951E + 43	1.0000058
6	3.73392905E + 43	3.73393117E + 43	0.9999994
7	1.71474218E + 44	1.71473453E + 44	1.0000045
8	8.43270714E + 41	8.43269136E + 41	1.0000019
9	4.65947847E + 42	4.65947050E + 42	1.0000017
10	1.16940744E + 41	1.16939498E + 41	1.0000107
11	5.95996723E + 42	5.95995793E + 42	1.0000016
12	8.80868593E + 40	8.80861947E + 40	1.0000075
13	8.77137865E + 41	8.77137533E + 41	1.0000004
14	1.44872160E + 43	1.44871775E + 43	1.0000027
15	3.09001312E + 41	3.09000648E + 41	1.0000022
16	6.56958283E + 43	6.56951477E + 43	1.0000104
17	8.16661377E + 40	8.16664440E + 40	0.9999962
18	5.43962621E + 40	5.43964802E + 40	0.9999960
19	2.02031274E + 41	2.02030609E + 41	1.0000033
20	5.45890832E + 40	5.45891507E + 40	0.9999988
21	5.75005089E + 41	5.75003179E + 41	1.0000033
22	8.96783127E + 43	8.96779937E + 43	1.0000036
23	3.27846110E + 43	3.27845684E + 43	1.0000013
24	1.06140068E + 42	1.06139886E + 42	1.0000017
25	6.83926724E + 44	6.83928425E + 44	0.9999975

Table V
Average Values

	10-Job	20-Job	100-Job	250-Job
H_1/F^*	1.001661	1.001465	—	—
H_2/F^*	1.000315	1.000037	—	—
H_1/H_2	1.001346	1.001428	1.0000149	1.00000233

Table V shows that Heuristic 2 performs slightly better than Heuristic 1. The average relative difference between the heuristics on the 10-job and the 20-job sets is less than 0.2%. Heuristic 1 performs better only in 7 out of the 100 problems (problem 8 in Table I, problems 3, 6, 17, 18, 20 and 25 in Table IV).

Heuristic 2 also performs better in terms of its rate of convergence. As seen in Table IV, the improvement in the average relative error as the number of jobs increases from 10 to 20 is higher for Heuristic 2. For large sized problems, however, the average relative difference between the heuristics decreases significantly: it is less than 0.002% for the 100-job set and less than 0.0003% for the 250-job set. In these large sized problems the shape of both heuristic sequences is almost a perfectly symmetric V and therefore they are very close to the optimal solution.

Figure 1, 2, 3 and 4 demonstrate graphically all the above. Figure 1 shows the solution of Heuristic 1 for a 24-job problem against the optimal solution. Figure 2 shows the solution of Heuristic 2 for the same problem.

The data for the problem are: $\alpha_1 = 0.49$, $\alpha_2 = 0.24$, $\alpha_3 = 0.09$, $\alpha_4 = 0.10$, $\alpha_5 = 0.75$, $\alpha_6 = 0.52$, $\alpha_7 = 0.83$, $\alpha_8 = 0.36$, $\alpha_9 = 0.42$, $\alpha_{10} = 0.24$, $\alpha_{11} = 0.48$, $\alpha_{12} = 0.50$, $\alpha_{13} = 0.13$, $\alpha_{14} = 0.71$, $\alpha_{15} = 0.43$, $\alpha_{16} = 0.96$, $\alpha_{17} = 0.35$, $\alpha_{18} = 0.66$, $\alpha_{19} = 0.22$, $\alpha_{20} = 0.51$, $\alpha_{21} = 0.65$, $\alpha_{22} = 0.48$, $\alpha_{23} = 0.47$, $\alpha_{24} = 0.82$.

The results are:

$$F^* = 26932.426$$
$$H_1 = 26945.517$$
$$H_2 = 26932.485$$
$$H_1/F^* = 1.0004860$$
$$H_2/F^* = 1.0000022$$
$$H_1/H_2 = 1.0004838.$$

These figures again indicate that Heuristic 2 performs slightly better than Heuristic 1.

Figures 3 and 4 illustrate the second heuristic's solution to problem 1 of Table III and to problem 1 of Table IV, respectively. (It is clear from the values of H_1/H_2 in the tables that the solutions of Heuristic 1 are very similar.) Notice how the shape of the solution approaches a perfectly symmetric V as the number of jobs increases.

Interestingly enough, we find that in many problems, a large number of jobs are scheduled differently

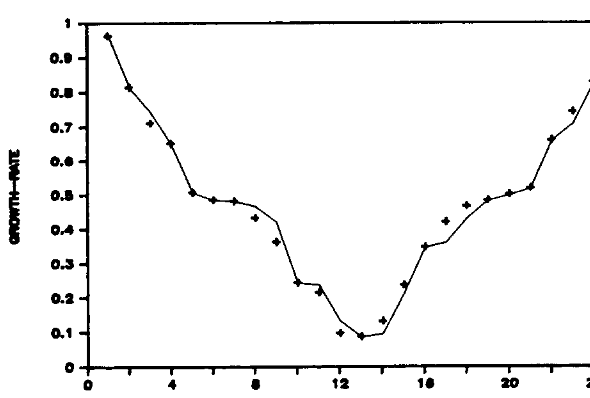

Figure 1. A 24-job problem (+ signs = heuristic solution H_1; solid line = optimal solution).

Copyright © 2001 All Rights Reserved

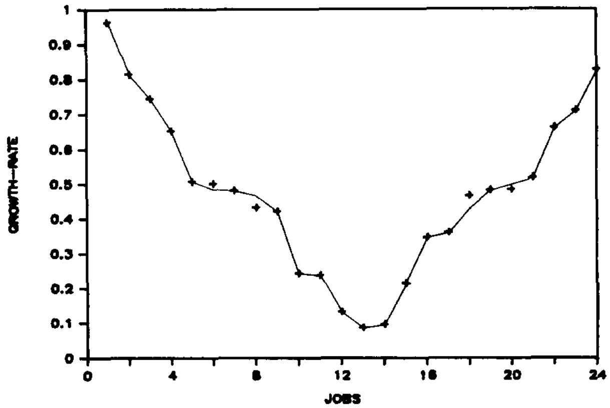

Figure 2. A 24-job problem (+ signs = heuristic solution H_2; solid line = optimal solution).

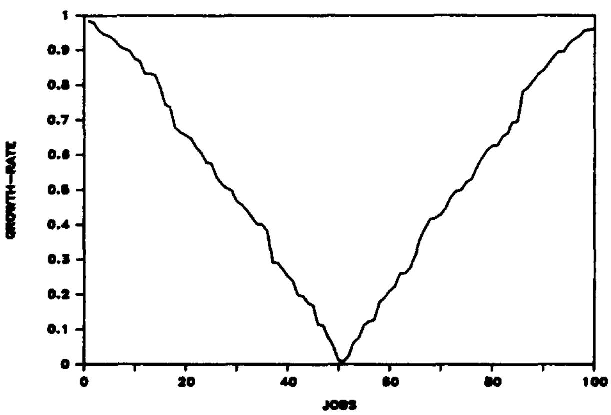

Figure 3. A 100-job problem (the solution of Heuristic 2).

by the optimal policy than by the heuristics. However, the resulting flow time of the heuristics is still extremely close to the optimal. For example, in the problem described in Figure 1, 10 out of 24 jobs are scheduled differently by the heuristic. The relative error is still less than 0.05%. Note, in addition, that $c(\pi^*) = c(\pi_1) = 0.13$ for this example, which means that the optimal and the heuristic sequences are equivalent in terms of closeness to a perfect V according to our measure. This shows that the "convergence" process of the heuristics solutions to a perfect V (and therefore to the optimal flow time), is faster than the convergence to the actual optimal sequence. It also verifies that the optimal solution is robust to changes in scheduling which maintain its general V-shape. Thus, any sequence that has a roughly symmetric V-shape guarantees very low flow time.

APPENDIX

Proof for Proposition 3

Consider the following set of growth rates:

$$\{\alpha_0, \alpha_1, \alpha_1, \alpha_2, \alpha_2, \ldots, \alpha_k, \alpha_k\};$$

$$\alpha_0 \geqslant \alpha_1 \geqslant \alpha_2 \geqslant \ldots \geqslant \alpha_k.$$

We will show that $(0, 1, 2, \ldots, k-1, k, k-1, \ldots, 2, 1)$ is the optimal sequence. (The total number of jobs in this case is $2k + 1$. The proof for the even number of jobs case is similar; the optimal sequence is $(0, 1, 2, \ldots, k-1, k, k-1, \ldots, 2, 1)$.)

In terms of the function g, the above is equivalent to the following: Given a set $A = \{a_1, a_1, a_2, a_2, \ldots, a_k, a_k\}$, we have to show that the minimizer of g is the sequence

$$\pi_0 = (1, 2, \ldots, k-1, k, k, k-1, \ldots, 2, 1).$$

(Recall that α_0 does not appear explicitly in g, and $a_i = 1 + \alpha_i$, $i = 1 \ldots k$.)

Since only V-shaped sequences are candidates for optimality it is enough to show that π_0 is the best within this set. Notice that any V-shaped sequence based on the set A which is not perfectly symmetric, must contain at least one "step," i.e., two adjacent jobs with identical growth rate.

Let

$$\pi_1 = (1, 2, \ldots, i-1, i, i, i+1, \ldots, k-1, k, k, k-1, \ldots, i+1, i-1, \ldots, 1).$$

We will show that $g(\pi_1) \geqslant g(\pi_0)$.

(We assume for simplicity of exposition that π_1

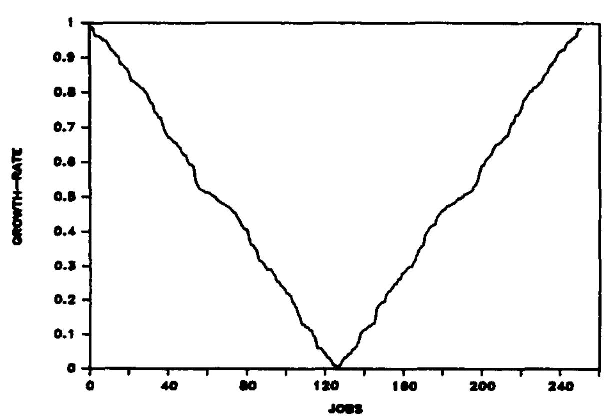

Figure 4. A 250-job problem (the solution of Heuristic 2).

Copyright © 2001 All Rights Reserved

contains a single step. Similar algebra is needed for the case of several steps.)

$$g(\pi_0) = a_1 + a_1 a_2 + a_2 + a_1 a_2 a_3 + a_2 a_3 + a_3 \\ \vdots \\ + a_1 a_2 a_3 \ldots a_{k-1} a_k a_k a_{k-1} \ldots a_2 a_1 + a_2 a_3 \ldots a_{k-1} a_k a_k a_{k-1} \ldots a_2 a_1 + \ldots + a_2 a_1 + a_1$$

$$g(\pi_1) = a_1 + a_1 a_2 + a_2 + a_1 a_2 a_3 + a_2 a_3 + a_3 \\ \vdots \\ + a_1 a_2 a_3 \ldots a_i + a_2 a_3 \ldots a_i + a_3 \ldots a_i + \ldots + a_i \\ + a_1 a_2 a_3 \ldots a_i a_i + a_2 a_3 \ldots a_i a_i + a_3 \ldots a_i a_i + \ldots + a_i a_i + a_i \\ \vdots \\ + a_1 a_2 \ldots a_i a_i \ldots a_{k-1} a_k a_k a_{k-1} \ldots a_{i+1} a_{i-1} \ldots a_2 a_1 \\ + a_2 \ldots a_i a_i \ldots a_{k-1} a_k a_k a_{k-1} \ldots a_{i+1} a_{i-1} \ldots a_2 a_1 + \ldots + a_2 a_1 + a_1.$$

First we identify identical terms in $g(\pi_0)$ and $g(\pi_1)$. Observe that the following block appears in both expressions:

$$a_1 + a_1 a_2 + a_2 + a_1 a_2 a_3 + a_2 a_3 + a_3 + \\ \vdots \\ + a_1 a_2 a_3 \ldots a_i + a_2 a_3 \ldots a_i + \ldots + a_{i-1} a_i + a_i.$$

Note that this expression is $g(a_1, a_2, \ldots, a_i)$. Similarly, the block $g(a_{i-1}, a_{i-2}, \ldots, a_1)$ appears in both expressions. In addition, $g(a_{i+1}, a_{i+2}, \ldots, a_k, a_k, a_{k-1}, \ldots, a_i)$ appears in $g(\pi_0)$, and $g(a_i, a_{i+1}, \ldots, a_k, a_k, a_{k-1}, \ldots, a_{i+1})$ appears in $g(\pi_1)$. By the symmetry of g (property 2) these expressions are equal. Other equal terms are the ones in π_0 that contain the product $X = a_{i+1} a_{i+2} \ldots a_{k-1} a_k a_k a_{k-1} \ldots a_i$. These terms are included in the block:

$$a_1 a_2 a_3 \ldots a_i X + a_2 a_3 \ldots a_i X + \ldots + X \\ + a_1 a_2 a_3 \ldots a_i X a_{i-1} + a_2 a_3 \ldots a_i X a_{i-1} + \ldots + X a_{i-1} \\ \vdots \\ + a_1 a_2 a_3 \ldots a_i X a_{i-1} a_{i-2} \ldots a_1 \\ + a_2 a_3 \ldots a_i X a_{i-1} a_{i-2} \ldots a_1 + \ldots + X a_{i-1} a_{i-2} \ldots a_1.$$

Similar terms with $Y = a_i a_{i+1} \ldots a_{k-1} a_k a_k a_{k-1} \ldots a_{i+1}$ replacing X appear in π_1. Since $X = Y$ these two blocks are equivalent.

Denote the total value of all common terms by Z. The remaining terms of $g(\pi_0)$ are included in two blocks:

$$g(\pi_0) - Z = B_1 + B_2$$

where

$$B_1 = a_i a_{i+1} + a_{i-1} a_i a_{i+1} + \ldots + a_1 a_2 \ldots a_i a_{i+1} + \\ a_i a_{i+1} a_{i+2} + a_{i-1} a_i a_{i+1} a_{i+2} + \ldots + a_1 a_2 \ldots a_i a_{i+1} a_{i+2} + \\ \vdots \\ a_i a_{i+1} \ldots a_k + a_{i-1} a_i a_{i+1} \ldots a_k + \ldots + a_1 a_2 \ldots a_i a_{i+1} \ldots a_k + \\ \vdots \\ a_i a_{i+1} \ldots a_k a_k a_{k-1} \ldots a_{i+1} \\ + a_{i-1} a_i a_{i+1} \ldots a_k a_k a_{k-1} \ldots a_{i+1} + \ldots \\ + a_1 a_2 \ldots a_i a_{i+1} \ldots a_k a_k a_{k-1} \ldots a_{i+1}.$$

$$B_2 = a_i a_{i-1} + a_i a_{i-1} a_{i-2} + \ldots + a_i a_{i-1} \ldots a_1 + \\ a_{i+1} a_i a_{i-1} + a_{i+1} a_i a_{i-1} a_{i-2} + \ldots + a_{i+1} a_i a_{i-1} \ldots a_1 + \\ \vdots$$

Copyright © 2001 All Rights Reserved

$$a_k \dots a_i a_{i-1} + a_k \dots a_i a_{i-1} a_{i-2} + \dots + a_k \dots a_i a_{i-1} \dots a_1 +$$

$$\vdots$$

$$a_{i+2} \dots a_k a_k a_{k-1} \dots a_i a_{i-1} + a_{i+2} \dots a_k a_k a_{k-1} \dots a_i a_{i-1} a_{i-2} + \dots + a_{i+2} \dots a_k a_k a_{k-1} \dots a_i a_{i-1} \dots a_1.$$

The remaining terms of $g(\pi_1)$ are included in two corresponding blocks:

$$g(\pi_1) - Z = B_1' + B_2'$$

where

$$B_1' = a_i a_i + a_{i-1} a_i a_i + \dots + a_1 a_2 \dots a_{i-1} a_i a_i +$$
$$a_i a_i a_{i+1} + a_{i-1} a_i a_i a_{i+1} + \dots + a_1 a_2 \dots a_{i-1} a_i a_i a_{i+1} +$$
$$\vdots$$
$$a_i a_i a_{i+1} \dots a_k + a_{i-1} a_i a_i a_{i+1} \dots a_k + \dots + a_1 a_2 \dots a_{i-1} a_i a_i a_{i+1} \dots a_k +$$
$$\vdots$$
$$a_i a_i a_{i+1} \dots a_k a_k a_{k-1} \dots a_{i+2} + a_{i-1} a_i a_i a_{i+1} \dots a_k a_k a_{k-1} \dots a_{i+2} + \dots + a_1 a_2 \dots a_{i-1} a_i a_i a_{i+1} \dots a_k a_k a_{k-1} \dots a_{i+2}.$$

$$B_2' = a_{i+1} a_{i-1} + a_{i+1} a_{i-1} a_{i-2} + \dots + a_{i+1} a_{i-1} a_{i-2} \dots a_1 +$$
$$a_{i+2} a_{i+1} a_{i-1} + a_{i+2} a_{i+1} a_{i-1} a_{i-2} + \dots + a_{i+2} a_{i+1} a_{i-1} a_{i-2} \dots a_1 +$$
$$\vdots$$
$$a_k \dots a_{i+1} a_{i-1} + a_k \dots a_{i+1} a_{i-1} a_{i-2} + \dots + a_k \dots a_{i+1} a_{i-1} a_{i-2} \dots a_1 +$$
$$\vdots$$
$$a_{i+1} \dots a_k a_k a_{k-1} \dots a_{i+1} a_{i-1} + a_{i+1} \dots a_k a_k a_{k-1} \dots a_{i+1} a_{i-1} a_{i-2} + \dots + a_{i+1} \dots a_k a_k a_{k-1} \dots a_{i+1} a_{i-1} a_{i-2} \dots a_1.$$

We compare the blocks line by line:

$$B_1' - B_1$$
$$= (a_i - a_{i+1})(a_i + a_{i-1} a_i + a_{i-2} a_{i-1} a_i + \dots a_1 a_2 \dots + a_i) \quad (1)$$
$$+ (a_i - a_{i+2})(a_i a_{i+1} + a_{i-1} a_i a_{i+1} + \dots + a_1 a_2 \dots a_{i+1}) \quad (2)$$
$$+ (a_i - a_{i+3})(a_i a_{i+1} a_{i+2} + a_{i-1} a_i a_{i+1} a_{i+2} + \dots + a_1 a_2 \dots a_{i+2}) \quad (3)$$
$$\vdots$$
$$+ (a_i - a_k)(a_i a_{i+1} \dots a_{k-1} + a_{i-1} a_i a_{i+1} \dots a_{k-1} + \dots + a_1 a_2 \dots ak-1) \quad (k-i)$$
$$+ (a_i - a_k)(a_i a_{i+1} \dots a_{k-1} a_k + a_{i-1} a_i a_{i+1} \dots a_{k-1} a_k + \dots + a_1 a_2 \dots a_{k-1} a_k) \quad (k-i+1)$$
$$+ (a_i - a_{k-1})(a_i a_{i+1} a_{i+2} \dots a_k a_k + a_{i-1} a_i a_{i+1} \dots a_k a_k + \dots + a_1 a_2 \dots a_k a_k) \quad (k-i+2)$$
$$\vdots$$
$$+ (a_i - a_{i+1})(a_i a_{i+1} a_{i+2} \dots a_k a_k \dots a_{i+2} + a_{i-1} a_i a_i a_{i+1} a_{i+2} \dots a_k a_k \dots a_{i+2} + \dots + a_1 a_2 \dots a_k a_k \dots a_{i+2}). \quad (2k-2i)$$

$$B_2' - B_2$$
$$= (a_{i+1} - a_i)(a_{i-1} + a_{i-1} a_{i-2} + \dots + a_{i-1} a_{i-2} \dots a_1) \quad (1)'$$
$$+ (a_{i+2} - a_i)(a_{i+1} a_{i-1} + a_{i+1} a_{i-1} a_{i-2} + \dots + a_{i+1} a_{i-1} \dots a_1) \quad (2)'$$
$$\vdots$$
$$+ (a_k - a_i)(a_{k-1} a_{k-2} \dots a_{i+1} a_{i-1} + a_{k-1} a_{k-2} \dots a_{i+1} a_{i-1} a_{i-2} + \dots + a_{k-1} a_{k-2} \dots a_{i+1} a_{i-1} \dots a_1) \quad (k-i)'$$
$$\vdots$$

Copyright © 2001 All Rights Reserved

$$+ (a_{i+1} - a_i)(a_{i+2}a_{i+3} \ldots a_k a_k \ldots a_{i+1}a_{i-1}$$
$$+ a_{i+2}a_{i+3} \ldots a_k a_k \ldots a_{i+1}a_{i-1}a_{i-2} + \ldots$$
$$+ a_{i+2}a_{i+3} \ldots a_k a_k \ldots a_{i+1}a_{i-1} \ldots a_1) \quad (2k - 2i)'$$

Denote the expression in the second parenthesis of line (j') by S_j:

$$(1)' = (a_{i+1} - a_i)S_1,\ (2)' = (a_{i+2} - a_i)S_2, \ldots.$$

Then it is easy to verify that

$$(1) + (1)' = (a_i - a_{i+1})[S_1(a_i - 1) + a_i] \geqslant 0$$

$$(2) + (2)' = (a_i - a_{i+2})[S_2(a_i - 1) + a_i a_{i+1}] \geqslant 0$$

$$\vdots$$

$$(k - i) + (k - i)' = (a_i - a_k)[S_{k-i}(a_i - 1) + a_i a_{i+1} \ldots a_{k-1}] \geqslant 0$$

$$\vdots$$

$$(2k - 2i) + (2k - 2i)' = (a_i - a_{i+1})[S_{2k-2i}(a_i - 1) + a_i a_{i+1} \ldots a_k a_k a_{k-1} \ldots a_{i+2}] \geqslant 0.$$

Therefore

$$g(\pi_1) - g(\pi_0) = (B_1' - B_1) + (B_2' - B_2) \geqslant 0,$$

and the inequality is strict for any nontrivial set $\{a_1, \ldots, a_k\}$ (i.e., $a_j > a_{j+1}$ for some j). This completes the proof.

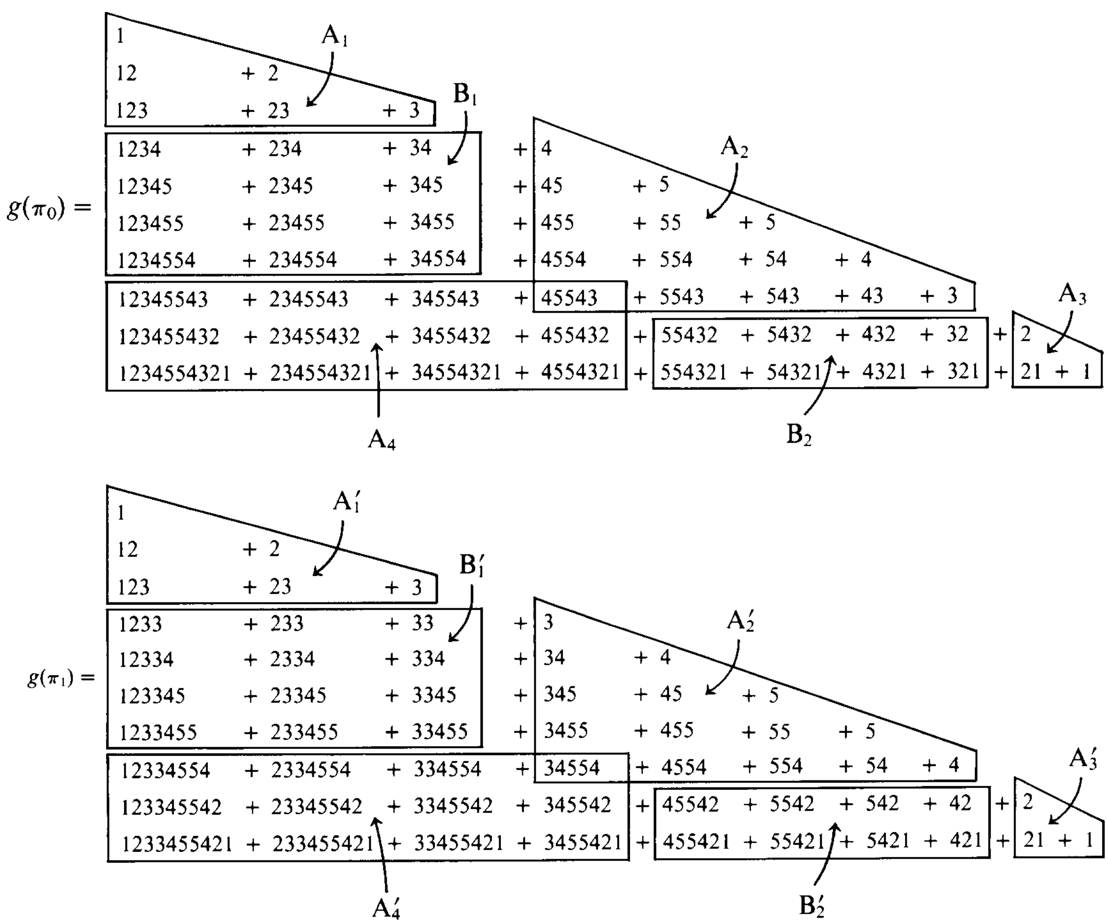

Figure 5. $g(\pi_0)$ and $g(\pi_1$.

Copyright © 2001 All Rights Reserved

Example. For a demonstration of the block decomposition of $g(\pi_0)$ and $g(\pi_1)$ that was used in the proof consider the following example with $k = 5$ and $i = 3$. The sequences to be compared are:

$$\pi_0 = (1, 2, 3, 4, 5, 5, 4, 3, 2, 1)$$

and

$$\pi_1 = (1, 2, 3, 3, 4, 5, 5, 4, 2, 1).$$

(For ease of exposition each a_i is denoted by its index i, e.g., the product $a_4a_5a_5$ is denoted by 455.) See Figure 5. We show that $A_1 = A'_1$, $A_2 = A'_2$, $A_3 = A'_3$, $A_4 = A'_4$, and $B'_1 + B'_2 \geq B_1 + B_2$.

ACKNOWLEDGMENT

I wish to thank Sid Browne for introducing the problem to me and for many helpful discussions. I also thank Awi Federgruen, Saul Lach, two anonymous referees and the associate editor for their most useful comments.

REFERENCES

Bagchi, U., Y. Chang and R. Sullivan. 1987. Minimizing Absolute and Squared Deviations of Completion Times With Different Earliness and Tardiness Penalties and a Common Due Date. *Naval Res. Logist. Quart.* **34,** 739–751.

Bagchi, U., R. S. Sullivan and Y. L. Chang. 1986. Minimizing Mean Absolute Deviation of Completion Times About a Common Due Date. *Naval Res. Logist. Quart.* **33,** 227–240.

Bagchi, U., R. S. Sullivan and Y. L. Chang. 1987. Minimizing Mean Squared Deviation of Completion Time About a Common Due Date. *Mgmt. Sci.* **33,** 894–906.

Baker, K. R., and G. D. Scudder. 1990. Sequencing With Earliness and Tardiness Penalties: A Review. *Opns. Res.* **38,** 22–38.

Browne, S., and U. Yechiali. 1990. Scheduling Deteriorating Jobs on a Single Processor. *Opns. Res.* **38,** 495–498.

Eilon, S., and I. G. Chowdhury. 1977. Minimizing Waiting-Time Variance in the Single Machine Problem. *Mgmt. Sci.* **23,** 567–575.

Emmons, H. 1987. Scheduling to a Common Due Date on Parallel Common Processors. *Naval Res. Logist. Quart.* **34,** 803–810.

Hall, N. G. 1986. Single- and Multiple-Processor Models for Minimizing Completion-Time Variance. *Naval Res. Logist. Quart.* **33,** 49–54.

Hall, N. G., and M. Posner. 1991. Weighted Deviation of Completion Times About a Common Due Date. *Opns. Res.* **39,** 836–846.

Kanet, J. J. 1981. Minimizing the Average Deviation of Job Completion-Times About a Common Due-Date. *Naval Res. Logist. Quart.* **28,** 643–651.

Merten, A. G., and M. E. Muller. 1972. Variance Minimization in Single Machine Sequencing Problems. *Mgmt. Sci.* **18,** 518–528.

Panwalker, S. S., M. L. Smith and A. Seidman. 1982. Common Due-Date Assignment to Minimize Total Penalty for the One Machine Scheduling Problem. *Opns. Res.* **30,** 391–399.

Pinedo, M. 1982. Minimizing the Expected Makespan in Stochastic Flow Shops. *Opns. Res.* **30,** 148–162.

Schrage, L. 1975. Minimizing the Time-in-System Variance for a Finite Jobset. *Mgmt. Sci.* **21,** 540–543.

Sundararaghavan, P. S., and M. U. Ahmed. 1984. Minimizing the Sum of Absolute Lateness in Single-Machine and Multi-Machine Scheduling. *Naval Res. Logist. Quart.* **31,** 325–333.

Copyright © 2001 All Rights Reserved