SCHOOL OF OPERATIONS RESEARCH
AND INDUSTRIAL ENGINEERING
COLLEGE OF ENGINEERING
CORNELL UNIVERSITY
ITHACA, NY 14853-7501

TECHNICAL REPORT NO. 919

August 1990

PERMUTATION VS. NON-PERMUTATION FLOW SHOP SCHEDULES

by

Chris N. Potts[1], David B. Shmoys[2]
David P. Williamson[3]

[1]University of Southampton

[2]This research was supported in part by NSF PYI Award CCR-89-96272 with matching support from UPS, IBM and Sun and by the Cornell Computational Optimization Project.

[3]Laboratory for Computer Science, 545 Technology Sq., Cambridge, MA 02139. This research was supported in part by a NSF graduate fellowship and Siemens Research Agreement of June 10, 1987 on Joint Research in Learning Systems.

Permutation vs. Non-Permutation Flow Shop Schedules

Chris N. Potts
University of Southampton

David B. Shmoys*
Cornell University

David P. Williamson†
MIT

Abstract

In studying the m-machine flow shop scheduling problem, it is common practice to focus attention on permutation schedules. We show that for the problem of minimizing the maximum completion time, this assumption can be a costly one, by exhibiting a family of instances for which the value of the best permutation schedule is worse than that of the true optimal schedule by a factor of more than $\sqrt{m}/2$.

Keywords: Flow-shop scheduling, permutation schedules, approximation algorithms

In the flow shop scheduling problem, there are n jobs and m machines, and each job j, $j = 1, 2, \ldots, n$, consists of m operations where the ith operation is to be performed, in order, by machine i for a specified time p_{ij}. In other words, each job j is first processed by machine 1, then machine 2, and so on, until it completes its processing by machine m; the time at which job j completes its last operation shall be denoted C_j. The objective is to find a nonpreemptive schedule that minimizes $C_{max} = \max_j C_j$, the maximum completion time of any job. This optimum shall be denoted C^*_{max}. The recent paper of Lawler, Lenstra,

*Current address: ORIE, Upson Hall, Cornell University, Ithaca, NY, 14853. This research was supported in part by NSF PYI Award CCR-89-96272 with matching support from UPS, IBM and Sun and by the Cornell Computational Optimization Project.

†Current address: Laboratory for Computer Science, 545 Technology Sq., Cambridge, MA, 02139. This research was supported in part by a NSF graduate fellowship and Siemens Research Agreement of June 10, 1987 on Joint Research in Learning Systems.

Rinnooy Kan & Shmoys [2] surveys many of the known results pertaining to this problem.

Permutation schedules constitute an important subclass of schedules, where the order in which each *machine* processes the jobs is identical for all machines. Most research on flow shop scheduling problems has focused on permutation schedules, due the relative combinatorial simplicity of schedules that can be specified simply by giving a permutation of the jobs. Unfortunately, this simplicity is bought at the price of drastically inferior schedules, and the purpose of this note is to study the worst-case behavior of the ratio of the maximum completion time of an optimal permutation schedule, denoted $C^*_{max}(\pi)$, to the optimum value C^*_{max}. If $\rho(\mathcal{I})$ denotes this ratio for an instance $\mathcal{I}$, we will prove that ρ is not bounded by any constant, by exhibiting a family of instances $\{\mathcal{I}_{2n} : n \in \boldsymbol{N}\}$ for which $\rho(\mathcal{I}_m) \geq \lceil \sqrt{m} + \frac{1}{2} \rceil / 2$, where the instance $\mathcal{I}_m$ involves m machines.

The instance $\mathcal{I}_{2n}$ is relatively easy to describe: there are n jobs, $2n$ machines, and

$$p_{ij} = \begin{cases} 1 & \text{if } i = j + n \text{ or } i = n + 1 - j \\ 0 & \text{otherwise.} \end{cases}$$

When $p_{ij} = 0$, this value should rather be interpreted as an arbitrarily small positive constant; in other words, each job must still be processed on each machine. It is easy to see that $C^*_{max} = 2$; since each job requires 2 units of processing, any schedule must take at least that long, and if the jobs are processed in the order $1, 2, \ldots, n$ on machines $1, 2, \ldots, n$, but then in the order $n, n-1, \ldots, 1$ on machines $n+1, n+2, \ldots, 2n$, all jobs are completed at the end of the second time unit. To prove the main result of this note, we need only prove a lower bound on the length of the optimal permutation schedule. The fundamental insight into analyzing the length of permutation schedules for these instances is the following easy fact:

Fact 1 For the instance $\mathcal{I}_{2n}$, $C_{max} = n + 1$ for the schedules given by either of the permutations $1, 2, \ldots, n$ or $n, n-1, \ldots, 1$.

Proof: The schedules are given in Figure 1. □

This simple fact will have important consequences. Consider an arbitrary permutation of the n jobs $j_1, j_2, \ldots j_n$ and let $j_{i_1}, j_{i_2}, \ldots, j_{i_s}$ be a subsequence of jobs such that $j_{i_1} < j_{i_2} < \cdots < j_{i_s}$. Note that the subinstance formed by this subset of jobs, along with machines $n+1-j_{i_s}, n+1-j_{i_{s-1}}, \ldots, n+1-j_{i_1}$ and $n+j_{i_1}, n+j_{i_2}, \ldots, n+j_{i_s}$ is isomorphic to I_{2s}. From this it follows that if there is an increasing (or decreasing) subsequence of length s in the permutation, then for this permutation schedule, $C_{max} \geq s+1$.

A well-known result of Erdös and Szekeres [1] states that in any sequence of k^2+1 distinct integers, there is either an increasing sequence of length $k+1$, or else there is a decreasing sequence of length $k+1$. Thus, this result can be combined with Fact 1 to prove an $\Omega(\sqrt{m})$ bound on $\rho(\mathcal{I}_m)$. In order to prove a tight bound, we will need to use some of the ideas in the proof of the Erdös-Szekeres result in a slightly stronger setting.

Theorem 1 For each instance $\mathcal{I}_{2n}$, $n = 1, 2, \ldots$, $C^*_{max}(\pi) \geq \lceil \sqrt{2n} + \frac{1}{2} \rceil$, and this bound is tight.

Proof: Let $j_1, j_2, \ldots, j_n$ denote a permutation of the jobs $1, 2, \ldots, n$. We will show that if the jobs are scheduled in this order, then the value of C_{max} for this schedule will be at least $\sqrt{2n} + \frac{1}{2}$, and since the value of C_{max} is an integer, this implies the claimed result. The proof will isolate a particular subsequence of jobs that will, by themselves, imply this lower bound.

Let ℓ denote the length of the longest increasing subsequence in $j_1, j_2, \ldots, j_n$, and let ℓ_i denote the length of the longest increasing subsequence that starts with the element j_i. Note that $\ell_i \in \{1, \ldots, \ell\}$. Consider the multiset $S = \{\ell_i : i = 1, \ldots, n\}$, and let m_k denote the number of times that k occurs in S. It is evident from the jobs of a longest increasing subsequence that $m_k > 0$ for $k = 1, \ldots, \ell$. For $k = 1, \ldots, \ell$, we will show that the (non-empty) subsequence containing all jobs j_i for which $\ell_i = k$ must complete processing no earlier than $k + m_k$. If we let k^* denote the value $k \in \{1, 2, \ldots, \ell\}$ for which $k + m_k$ is

maximized, we will also show that $k^* + m_{k^*} \geq \lceil \sqrt{2n} + \frac{1}{2} \rceil$. By combining these two facts, we obtain the claimed lower bound.

Let $j_{i_1}, j_{i_2}, \ldots, j_{i_s}$ denote the subsequence of jobs j_i for which $\ell_i = k$. Note that $s = m_k$. If j_u and j_v are two jobs of this subsequence with $u < v$, it is easy to see that $j_u < j_v$ implies that $\ell_u > \ell_v$, since j_u can be placed at the head of the longest increasing sequence starting with j_v. Thus, $\ell_{i_1} = \ell_{i_2} = \cdots = \ell_{i_s}$ implies that $j_{i_1}, j_{i_2}, \ldots, j_{i_s}$ forms a decreasing sequence. Note that job j_{i_1} must be processed for one time unit on machine $n + 1 - j_{i_1}$, and after that j_{i_2} must be processed for one time unit on machine $n + 1 - j_{i_2}$, and so forth, so that job j_{i_s} completes its processing on machine n no earlier than after $s = m_k$ units of time. But now consider the increasing subsequence of jobs of length k that starts with j_{i_s}; call this sequence $j_{i_s}, j_{i_{s+1}}, \ldots, j_{i_{s+k-1}}$. This sequence must now be processed on machines $n+1, \ldots, 2n$. Once again, first j_{i_s} must be processed for one time unit on machine $n + j_{i_s}$, and this must be followed by $j_{i_{s+1}}$ on machine $n + j_{i_{s+1}}$, and so forth, so that $j_{i_{s+k-1}}$ is completed on machine $2n$ at least k time units later. Thus, the subsequence requires in total at least $k + m_k$ time units to complete processing.

To prove the second fact, suppose that $k + m_k \leq r$, for each $k = 1, \ldots, \ell$. Then $\sum_{k=1}^{\ell}(k+m_k) \leq \ell \cdot r$. But note that $\sum_{k=1}^{\ell} k = \ell(\ell+1)/2$ and $\sum_{k=1}^{\ell} m_k = n$. Thus, $r \geq n/\ell + (\ell+1)/2$. Applying elementary calculus, we see that the right-hand side achieves its minimum when $\ell = \sqrt{2n}$, and so $r \geq \sqrt{2n} + \frac{1}{2}$. Since $k^* + m_{k^*}$ is an integer, we obtain the desired inequality.

To see that this bound is tight, consider the schedule given by the permutation $1, 3, 2, 6, 5, 4, 10, 9, 8, 7, \ldots$; the pattern of increasingly longer blocks of decreasing sequences should be clear from this prefix. For $n = 10$, this schedule is given in Figure 2. Extrapolating from this example, we see that if $k(k-1)/2 < n \leq k(k+1)/2$, then this schedule for $\mathcal{I}_{2n}$ has $C_{max} = k+1$, which equals the bound claimed in the theorem. □

The algorithm of Röck and Schmidt [4] proves that $\rho(\mathcal{I}) \leq \lceil m/2 \rceil$ for any

instance $\mathcal{I}$. It remains an interesting open question to determine whether there is a family of instances with an asymptotically larger ratio than $O(\sqrt{m})$ or, from the opposite (and more interesting) point of view, whether there is an algorithm that always delivers a permutation schedule with $C_{max} \leq O(\sqrt{m})C^*_{max}$.

References

[1] P. Erdös and A. Szekeres. A combinatorial problem in geometry. *Compositio Mathematica*, 2:463–470, 1935.

[2] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, and D.B. Shmoys. *Sequencing and Scheduling: Algorithms and Complexity.* Technical Report Report BS-R8909, Centre for Mathematics and Computer Science, 1989.

[3] H. Röck and G. Schmidt. *Machine Aggregation Heuristics in Shop Scheduling.* Technical Report Berich 82-11, Fachbereich 20 Mathematik, Technische Universität Berlin, 1982.

[4] H. Röck and G. Schmidt. Machine aggregation heuristics in shop-scheduling. *Methods of Oper. Res.*, 45:303–314, 1983.

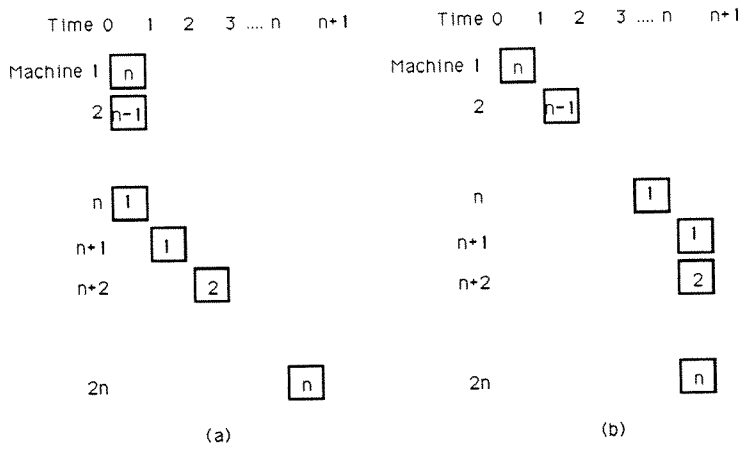

Figure 1: (a) The permutation schedule $1, 2, \ldots, n$; (b) the permutation schedule $n, n-1, \ldots, 1$.

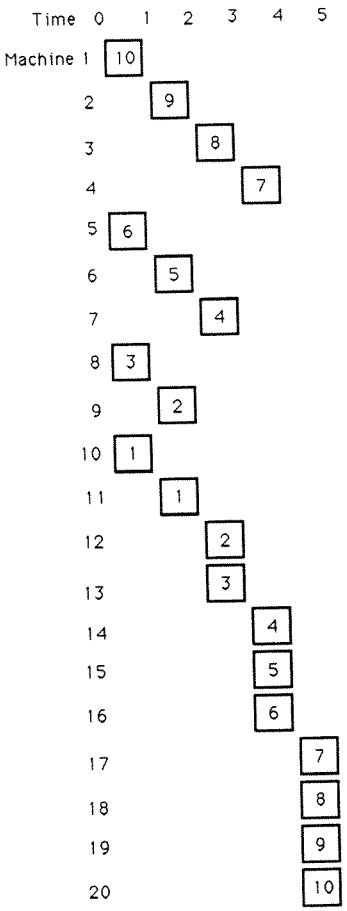

Figure 2: A good permutation schedule.