Precedence Constrained Scheduling to Minimize Sum of Weighted Completion Times on a Single Machine

CHANDRA CHEKURI* RAJEEV MOTWANI †

Abstract

We consider the problem of scheduling a set of jobs on a single machine with the objective of minimizing sum of weighted completion times. The problem is NP-hard when there are precedence constraints between jobs [15]. We provide an efficient combinatorial 2-approximation algorithm for this problem. In contrast to our work, earlier approximation algorithms [12] achieving constant factor approximations are based on solving a linear programming relaxation of the problem. We also show that the linear ordering relaxation of Potts [20] has an integrality gap of 2.

1 Introduction

We consider the following scheduling problem. We are given a set of jobs $J_1, J_2, \ldots, J_n$ where each job J_i has a processing time p_i and a weight w_i. Jobs have precedence constraints between them that are specified in the form of a directed acyclic graph. If $i \prec j$, J_j cannot be scheduled before J_i completes. The objective is to find a *non-preemptive* schedule of the jobs on a single machine (or equivalently an ordering), to minimize $\sum_i w_i C_i$, where C_i is the completion time of J_i in the schedule. Scheduling problems with the objective of minimizing $\sum_j w_j C_j$ have been studied extensively in operations research. Computer

*Bell Labs, 600 Mountain Ave, Murray Hill, NJ 07974. E-mail: `chekuri@research.bell-labs.com`. This work done while author was at Stanford University where he was primarily supported by an IBM Cooperative Fellowship. Additional supported by NSF Award CCR-9357849, with matching funds from IBM, Mitsubishi, Schlumberger Foundation, Shell Foundation, and Xerox Corporation.

†Computer Science Department, Rm 474, Gates Bldg 4B, Stanford, CA 94305-9045. E-mail: `rajeev@cs.stanford.edu`. Supported by an Alfred P. Sloan Research Fellowship, an IBM Faculty Partnership Award, an ARO MURI Grant DAAH04-96-1-0007, and NSF Young Investigator Award CCR-9357849, with matching funds from IBM, Mitsubishi, Schlumberger Foundation, Shell Foundation, and Xerox Corporation.

science applications include profile driven instruction scheduling in VLIW and superscalar compilers [3].

Our problem is NP-hard if we permit arbitrary precedence constraints on the jobs [9, 15]. It is polynomial-time solvable when the precedence graph is a forest [13], or a generalized series-parallel graph [15, 1]. The best known approximation algorithm for the general DAG case until recently had a ratio of $O(\log n \log L)$ where $L = \sum_i p_i$ is the sum of the execution times of the jobs [22]. Recently, Hall et al. [12] gave constant factor approximations using linear programming relaxations. It is interesting to note that several different formulations give the same bound of 2 [12]. Our analysis might give some insight as to why this is the case.

A more general version of the problem is to schedule the jobs on a set of m identical machines. For $m \geq 2$, the problem is NP-hard even without precedence constraints, unless the weights are all identical in which case it is polynomial-time solvable; on the other hand, the problem is strongly NP-hard even when all weights are identical and the precedence graph is a collection of chains [6]. An approximation ratio of 5.33 is achievable even if there are release times on the jobs [2]. A recent result by Munier et al. [19] gives a 4 approximation. Chekuri et al. [4] gave an algorithm which converts an α approximate schedule for a single machine problem into a $2\alpha+2$ approximate schedule for the multiple machine problem. This algorithm, based on a novel modification to list scheduling, is combinatorial and efficient.

In this paper, we give an efficient combinatorial 2 approximation algorithm for the single machine problem which matches the best ratio achieved in [12]. The advantage of our algorithm is twofold. First, the algorithms in [12] are based on solving linear programming relaxations while ours is based on solving a minimum cut, and hence is more efficient. Second, combining our algorithm with the conversion algorithm in [4], we get an efficient combinatorial algorithm for the multiple machine case as well. Margot, Queyranne, and Wang [17] have independently obtained the same 2 approximation algorithm that we present in this paper.

In a recent paper, Chudak and Hochbaum [5] show a half integral linear programming relaxation for the same problem which achieves an approximation ratio of 2. The half-integral program can be solved using a minimum cut computation, and thus yields a combinatorial algorithm. However the running time obtained is worse than that of our algorithm by a factor of at least $\Omega(n)$. Their relaxation is a slight modification of Potts's linear ordering relaxation [20]. Hall et al. [12] showed that Potts's relaxation is feasible for their completion time relaxation, and hence provides a 2 approximation. Though factor 2 integrality gaps have been demonstrated for both the completion time relaxation and the time indexed relaxation [12], no such gap has been shown for the linear ordering relaxation. In this paper we show a factor 2 integrality gap for the linear ordering relaxation. Our example shows the same gap for Queyranne and Wang's formulation [21] as well. Surprisingly the instance on which we show the gap uses expander graphs.

2 Preliminaries and Notation

Let $G = (V, E)$ denote the precedence graph where V is the set of jobs. We will use jobs and vertices interchangeably. We say that i precedes j, denoted by $i \prec j$, if and only if there is a path from i to j in G. For any vertex $i \in V$, let G_i denote the subgraph of G induced by the set of vertices preceding i.

Definition 1 *The* rank *of a job J_i denoted by r_i is defined as $r_i = p_i/w_i$. Similarly, the* rank *of a set of jobs A denoted by $r(A)$ is defined as $r(A) = p(A)/w(A)$, where $p(A) = \sum_{J_i \in A} p_i$ and $w(A) = \sum_{J_i \in A} w_i$.*

Definition 2 *A subdag G' of G is said to be* precedence closed *if for every job $J_i \in G'$, G_i is a subgraph of G'.*

The rank of a graph is simply the rank of its node set.

Definition 3 *We define G^* to be a precedence-closed subgraph of G of minimum rank, i.e., among all precedence-closed subgraphs of G, G^* is of minimum rank.*

Note that G^* could be the entire graph G.

3 A Characterization of the Optimal Schedule

Smith's rule for a set of independent jobs states that there is an optimal schedule that schedules jobs in non-decreasing order of their ranks. We generalize this rule for the case of precedence constraints in a natural way. A version of the following theorem was proved by Sydney in 1975 [24] but the authors of this paper rediscovered it. We present our own proof for the sake of completeness.

Definition 4 *A* segment *in a schedule S is any set of jobs that are scheduled consecutively in S.*

Theorem 1 *There exists an optimal schedule for G in which an optimal schedule for G^* occurs as a segment that starts at time zero.*

Proof: The theorem is trivially true if G^* is the same as G. We consider the case when G^* is a proper subdag of G. Suppose the statement of the theorem is not true. Let S be some optimal schedule for G in which G^* does not occur as a segment that starts at time zero. For $k \geq 1$, let $A_1, A_2, \ldots, A_k$ be the segments of G^* in S, in increasing order of starting times. For $i \geq 2$ let the segment between A_{i-1} and A_i be denoted by B_i and let B_1 be the segment before A_1 that starts at time zero. For convenience we assume that B_1 is non-empty (we can always use a dummy segment with $p(B_1) = w(B_1) = 0$). Let $r(G^*) = \alpha$ and for $1 \leq j$ let B^j denote the union of the segments $B_1, B_2, \ldots, B_j$. For $1 \leq j$ it follows, from the definition of G^*, that $r(B^j) \geq \alpha$, for otherwise the union of B^j and G^* would be precedence closed and have rank less than α. Let A^j similarly denote the union of the segments $A_1, A_2, \ldots, A_j$. We also claim that $r(A^k - A^j) \leq \alpha$ for otherwise $r(A^j) < \alpha$.

Let S' be the schedule formed from S by moving all the A_i's ahead of B_j's while preserving their order within themselves. The schedule S' is legal since G^* is precedence closed. Let Δ denote the difference in the sum of weighted completion times of S and S'. We will show that $\Delta \geq 0$ which will complete the proof. While comparing the two schedules, we can ignore the contribution of the jobs that come after A_k since their status remains the same in S'. Let $\Delta(A_j)$ and $\Delta(B_j)$ denote the difference in weighted completion time of A_j and B_j respectively in S and S'. Therefore $\Delta = \sum_{j \leq k} \Delta(A_j) + \sum_{j \leq k} \Delta(B_j)$. It is easy to see that

$$\Delta(A_j) = w(A_j) p(B^j)$$

and

$$\Delta(B_j) = -w(B_j) p(A^k - A^{j-1}).$$

From our earlier observations on $r(B^j)$ and $r(A^k - A^j)$ we have $p(B^j) \geq \alpha w(B^j)$ and $p(A^k - A^j) \leq \alpha w(A^k - A^j)$. Therefore

$$\begin{aligned}
\Delta &= \sum_{j \leq k} \Delta(A_j) + \sum_{j \leq k} \Delta(B_j) \\
&= \sum_{j \leq k} w(A_j) p(B^j) - \sum_{j \leq k} w(B_j) p(A^k - A^{j-1}) \\
&\geq \alpha \sum_{j \leq k} w(A_j) w(B^j) - \alpha \sum_{j \leq k} w(B_j) w(A^k - A^{j-1}) \\
&= \alpha \sum_{j \leq k} w(A_j) \sum_{i \leq j} w(B_j) - \alpha \sum_{j \leq k} w(B_j) \sum_{i \geq j} w(A_i) \\
&= 0.
\end{aligned}$$

The third inequality above follows from our observations about $r(B^j)$ and $r(A - A^j)$ and the last equality follows from a simple change in the order of summation. □

When G^* is the same as G, Theorem 1 does not help in decomposing the problem.

4 A 2 approximation

Theorem 1 suggests the following natural algorithm. Given G, compute G^* and schedule G^* and $G - G^*$ recursively. It is not a priori clear that G^* can be computed in polynomial time, however we will reduce this problem to that of computing a maximum flow in an associated graph. The second and more important problem that needs to be solved before we have an algorithm is take care of the case when G^* is the same as G. We have to settle for an approximation in this case, for otherwise we would have a polynomial time algorithm to compute the optimal schedule.

The following lemma helps us in taking care of the case when $G^* = G$.

Lemma 1 *If G^* is the same as G,* $\text{OPT} \geq w(G) \cdot p(G)/2$.

Proof: Let $\alpha = r(G)$. Suppose S is an optimal schedule for G. Without loss of generality assume that the ordering of the jobs in S is $J_1, J_2, \ldots, J_n$. For any j, $1 \leq j \leq n$, observe that $C_j = \sum_{1 \leq i \leq j} p_i \geq \alpha \sum_{1 \leq i \leq j} w_i$. This is because the set of jobs $J_1, J_2, \ldots, J_j$ form a precedence closed subdag, and from our assumption on G^* it follows that $\sum_{i \leq j} p_j / \sum_{i \leq j} w_i \geq \alpha$. We bound the value of the optimal schedule as follows.

$$
\begin{aligned}
\text{OPT} &= \sum_j w_j C_j \\
&\geq \sum_j w_j \sum_{i \leq j} \alpha w_i \\
&= \alpha \left(\sum_j w_j^2 + \sum_{i<j} w_i w_j \right) \\
&= \alpha \left((\sum_j w_j)^2 - \sum_{i<j} w_i w_j \right) \\
&\geq \alpha \left(w(G)^2 - w(G)^2/2 \right) \\
&= \alpha w(G)^2/2 \\
&= w(G) p(G)/2
\end{aligned}
$$

The last equality is true because $r(G^*) = r(G) = p(G)/w(G) = \alpha$. □

Lemma 2 *Any feasible schedule with no idle time has a weighted completion time of at most $w(G) \cdot p(G)$.*

Proof: Clear. □

Theorem 2 *If G^* for a graph can be computed in time $O(T(n))$, then there is a 2 approximation algorithm for computing the minimum weighted completion time schedule that runs in time $O(nT(n))$.*

Proof: Given G, we compute G^* in time $O(T(n))$. If G^* is the same as G we schedule G arbitrarily and Lemmas 1 and 2 guarantee that we have a 2 approximation. If G^* is a proper subdag we recurse on G^* and $G - G^*$. From Theorem 1 we have $\textsc{opt}(G) = \textsc{opt}(G^*) + p(G^*) \cdot w(G - G^*) + \textsc{opt}(G - G^*)$. Inductively if we have 2 approximations for G^* and $G - G^*$ it is clear that we have a 2 approximation of the overall schedule. Now we establish the running time bound. We observe that $G^{**} = G^*$, therefore it suffices to recurse only on $G - G^*$. It follows that we make at most n calls to the routine to compute G^* and the bound follows. □

All that remains is to show how to compute G^* in polynomial time.

4.1 Computing G^*

An algorithm to compute G^* using a maximum flow computation is presented in Lawler's book [16]. We describe the algorithm, its proof, and some recent running time improvements for the sake of completeness. To compute G^* we consider the more general problem of finding a subdag of rank at most $\lambda > 0$, if one exists. We reduce the latter problem to the problem of computing an s-t mincut in an associated graph. The following defines the associated graph.

Definition 5 *Given a dag $G = (V, E)$, and a real number $\lambda > 0$, we define a capacitated directed graph $G_\lambda = (V \cup \{s, t\}, E', c)$ where the edge set E' is defined by $E' = \{(s, i), (i, t) \mid 1 \leq i \leq n\} \cup \{(i, j) \mid j \prec i\}$ and the capacities are defined by*

$$c(e) = \begin{cases} p_i & \text{if } e = (i, t) \\ \lambda w_i & \text{if } e = (s, i) \\ \infty & \text{otherwise} \end{cases}$$

Lemma 3 *Given a dag G, there is a subdag of rank at most λ if and only if the s-t mincut in G_λ is at most $\lambda w(G)$. If (A, B) is a cut whose value is bounded by $\lambda w(G)$, $r(A - \{s\}) \leq \lambda$ and $A - \{s\}$ is precedence closed in G.*

Proof: Let (A, B) be an s-t cut in G_λ whose value is bounded by $\lambda w(G)$. We first claim that $A - \{s\}$ is precedence closed in G. Suppose not. Then there is a pair of vertices (i, j)

such that $i \prec j$ and $j \in A$ and $i \notin A$. But then $c(j,i) = \infty$ which is a contradiction since $c(A,B) \leq \lambda w(G)$. From this fact and the definition of G_λ, it follows that

$$\begin{aligned} c(A,B) &= \sum_{i \in A} p_i + \sum_{i \notin A} \lambda w_i \\ &= \sum_{i \in A} (p_i - \lambda w_i) + \sum_{i \in V} \lambda w_i \\ &= \sum_{i \in A} (p_i - \lambda w_i) + \lambda w(G) \end{aligned}$$

Since $c(A,B) \leq \lambda w(G)$ it follows that $\sum_{i \in A}(p_i - \lambda w_i) \leq 0$ which implies that $r(A - \{s\}) \leq \lambda$. Using similar arguments as above, a precedence closed subdag A in G whose rank is less than λ induces a cut of value at most $\lambda w(G)$ in G_λ. □

Lemma 4 *G^* can be computed in strongly polynomial time $O(n^3)$, or in time $O(n^{8/3} \log U)$ where $U = \max_i(p_i + w_i)$.*

Proof: Computing the minimum cut in G_λ for each $\lambda > 0$ can be viewed as a parametric maxflow computation. There are at most n values of λ for which the minimum cut changes in the graph. Gallo et al. [7] show that it is possible to obtain all the distinct values of λ in time to do one maximum flow computation using the push-relabel algorithm. Goldberg and Tarjan's [11] push-relabel algorithm runs in $O(nm \log(n^2/m))$ time. Recently, Goldberg and Rao [10] improved the maximum flow running time to $O(\min\{n^{2/3}, m^{1/2}\} m \log(n^2/m) \log U)$ where U is the maximum capacity, and also showed that their bound applies for the parametric flow techniques of Gallo et al. [7]. The associated graph we construct has $\Omega(n^2)$ edges, therefore the claimed bounds follow. □

5 Integrality Gap of the Linear Ordering Relaxation

In this section we show that the linear ordering relaxation of Potts [20] has a factor 2 integrality gap. The gap also applies to the half integral formulation of Chudak and Hochbaum [5] which is a slight modification of the Potts's formulation. We first describe the linear ordering relaxation of Potts. For each pair of jobs i and j there is a $\{0,1\}$-variable δ_{ij} that is 1, if i is scheduled before j, and 0 otherwise. Either i is scheduled before j or vice versa, therefore

$$\delta_{ij} + \delta_{ji} = 1, \qquad 1 \leq i < j \leq n. \tag{1}$$

Precedence constraints imply that

$$\delta_{ij} = 1, \qquad i \prec j. \tag{2}$$

Transitive relations in a feasible schedule are captured by the following set of inequalities. They state that if i is scheduled before j and j is scheduled before k, then i is scheduled before k.

$$1 + \delta_{ik} \geq \delta_{ij} + \delta_{jk}, \qquad 1 \leq i, j, k \leq n, i \neq j \neq k \neq i. \tag{3}$$

The completion time of job j, indicated by the variable C_j, is given by

$$C_j = p_j + \sum_{k \neq j} \delta_{kj} \, p_k, \qquad 1 \leq j \leq n. \tag{4}$$

The linear ordering relaxation is simply

$$\begin{gathered} \min \sum_j w_j C_j \\ \text{subject to (1)–(4)} \\ \delta_{ij} \in \{0, 1\}. \end{gathered}$$

Potts showed that the above is a complete formulation of the single machine scheduling problem. We obtain a linear relaxation by replacing the integrality constraints on δ_{ij} by the following inequalities.

$$\delta_{ij} \geq 0, \qquad 1 \leq i \neq j \leq n. \tag{5}$$

Chudak and Hochbaum obtained a half integral formulation by replacing the transitive inequalities (3) by the following inequalities,

$$\delta_{ki} \leq \delta_{kj}, \qquad \text{if } i \prec j, k \neq j, k \neq i. \tag{6}$$

Now we give an instance of the scheduling problem for which the integrality gap of the above formulation is a factor of 2. We use a certain family of strongly expanding graphs. For n sufficiently large there exists an undirected bipartite graph $G = (L, R, E)$ such that

- $|L| = |R| = n$.
- Every vertex in R has degree $n^{3/4}$, and every vertex in L has degree at most $3n^{3/4}$.
- Every subset of $n^{3/4}$ vertices in R has at least $n - n^{3/4}$ neighbors in L.

The existence of such a graph can be shown by the probabilistic method (see Problem 5.5 in [18]). Let the vertices in L be numbered 1 to n, and those in R, $(n+1)$ to $2n$. We construct an instance of the scheduling problem as follows. For each vertex i in G, we have a corresponding job J_i, for a total of $2n$ jobs. For each edge $(i, j) \in E$ where $i \in L$ and

$j \in R$, we add the precedence constraint $i \prec j$. We set $p_i = 1$ for $1 \le i \le n$ and $p_i = 0$ for $(n+1) \le i \le 2n$. The weights are defined by $w_i = 1 - p_i$. Let I be the instance obtained. Two jobs i and j are unrelated, denoted by $i \prec\succ j$, if neither $i \prec j$ nor $j \prec i$ is true.

Lemma 5 *Setting $\delta_{ij} = 1$ if $i \prec j$ and $\delta_{ij} = 1/2$ if $i \prec\succ j$ gives a feasible solution to the linear ordering relaxation for I.*

Proof: It is easy to verify that equations (1)-(3) are satisfied by the stated assignment.

□

Lemma 6 *The optimum value of the linear ordering relaxation for I is at most $(n^2 + n^{7/4})/2$.*

Proof: We will show that the value of the relaxation for the feasible schedule obtained by setting the δ_{ij} as in Lemma 5 has a value at most $(n^2 + n^{7/4})/2$. We can ignore jobs $1, \ldots, n$ since their weights are 0. Consider a job $j > n$. We have

$$\begin{aligned} C_j &= p_j + \sum_{k \ne j} \delta_{kj}\, p_k \\ &= \sum_{k \prec j, k \le n} 1 + \sum_{k \prec\succ j, k \le n} 1/2 \\ &= n^{3/4} + (n - n^{3/4})/2 \\ &= (n + n^{3/4})/2. \end{aligned}$$

Summing up gives

$$\begin{aligned} \sum_j w_j C_j &= \sum_{j > n} C_j \\ &= \sum_{j > n} (n + n^{3/4})/2 \\ &= (n^2 + n^{7/4})/2. \end{aligned}$$

This proves the lemma. □

Lemma 7 *Any valid schedule for I has a sum of weighted completion time of at least $(n^2 - o(n^{7/4}))$.*

Proof: Let S be any valid schedule for I. Assume without loss of generality that the jobs $(n+1)$ to $2n$ are ordered in increasing order of completion times in S. We claim that $C_{(n+n^{3/4})} \ge (n - n^{3/4})$. To prove this consider the jobs $(n+1)$ to $(n + n^{3/4})$. Let

$A = \{i \mid i \leq n, i \prec j$ for some $(n+1) \leq j \leq (n+n^{3/4})\}$. A is the set of all predecessors of jobs $(n+1), \ldots, (n+n^{3/4})$. From the properties of the expander graph from which I was constructed, $|A| \geq (n - n^{3/4})$. Each job in A is completed before $C_{(n+n^{3/4})}$. The claim follows. Since $C_j \geq (n - n^{3/4})$ for all $(n+n^{3/4}) \leq j \leq 2n$ the lemma is proved. □

The following theorem follows from Lemmas 6 and 7.

Theorem 3 *The integrality gap of the linear ordering relaxation on I is $2 - o(1)$.*

6 Conclusions

Several different formulations and algorithms give a 2 approximation for the scheduling problem considered in this paper, and all of them have instances on which their guarantee is tight. An obvious open problem is to obtain an algorithm with an improved approximation ratio. Interestingly, no hardness of approximation results are known for the problem. We conjecture that the problem is NP-hard to approximate within some absolute constant factor (Max-SNP hard). Our results in this paper might help in addressing the above questions. Theorem 1 shows that it is sufficient to improve the approximation ratio for the case when $G^* = G$. In Section 5 we used expanders to construct specific instances with a factor of 2 integrality gap for the linear ordering relaxation of Potts [20]. We believe that understanding the role of expanders in such instances is crucial for both improving the approximation ratio and proving hardness of approximation results.

References

[1] D. Adolphson. Single machine job sequencing with precedence constraints. *SIAM Journal on Computing*, 6:40–54 (1977).

[2] S. Chakrabarti, C. A. Phillips, A. S. Schulz, D. B. Shmoys, C. Stein, and J. Wein. Improved scheduling algorithms for minsum criteria. In *Proceedings of the 23rd ICALP*, pages 646–57. Springer Verlag, 1996.

[3] C. Chekuri, R. Johnson, R. Motwani, B. K. Natarajan, B. R. Rau, and M. Schlansker. Profile-driven instruction level parallel scheduling with applications to super blocks. In *Proceeding of the 29th Annual International Symposium on Microarchitechture*, pages 58–67, 1996.

[4] C. Chekuri, R. Motwani, B. Natarajan, and C. Stein. Approximation techniques for average completion time scheduling. In *Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 609–618, 1997.

[5] F. Chudak and D. Hochbaum. A half-integral linear programming relaxation for scheduling precedence-constrained jobs on a single machine. *(submitted)*, August 1997.

[6] J. Du, J.Y.T. Leung, and G.H. Young. Scheduling chain structured tasks to minimize makespan and mean flow time. *Information and Computation*, 92:219–236 (1991).

[7] G. Gallo, M.D. Grigoriadis, and R. Tarjan. A fast parametric maximum flow algorithm and applications. *SIAM J. on Comput.*, 18:30–55, 1989.

[8] M.R. Garey. Optimal task sequencing with precedence constraints. *Discrete Mathematics*, 4:37–56 (1973).

[9] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness,* Freeman, San Francisco (1979).

[10] A.V. Goldberg and S. Rao. Beyond the Flow Decomposition Barrier. *Proceedings of the 38th Annual IEEE Symposium on Foundations of Computer Science*, 2–11 (1997).

[11] A.V. Goldberg and R.E. Tarjan. A New Approach to the Maximum Flow Problem. *J. Assoc. Comput. Mach.*, 35:921–940 (1988).

[12] L.A. Hall, A.S. Schulz, D.B. Shmoys, and J. Wein. Scheduling to minimize average completion time: Offline and online algorithms. *Math. of Operations Research*, 22:513–544 (1997).

[13] W.A. Horn. Single-machine job sequencing with treelike precedence ordering and linear delay penalties. *SIAM Journal of Applied Mathematics*, 23:189–202 (1972).

[14] T.C. Hu. Parallel sequencing and assembly line problems. *Operations Research*, 9:841–848 (1961).

[15] E.L. Lawler. Sequencing jobs to minimize total weighted completion time. *Annals of Discrete Mathematics*, 2:75–90 (1978).

[16] E.L. Lawler. *Combinatorial Optimization.* Holt, Rinehart, and Winston (1976).

[17] F. Margot, M. Queyranne, and Y. Wang. Personal communication, August 1997.

[18] R. Motwani and P. Raghavan. *Randomized Algorithms.* Cambridge University Press (1995).

[19] A. Munier, M. Queyranne, and A. S. Schulz. Approximation bounds for a general class of precedence constrained parallel machine scheduling problems. In R. E. Bixby, E. A. Boyd, and R. Z. Ríos-Mercado, editors, *Integer Programming and Combinatorial Optimization*, volume 1412 of *Lecture Notes in Computer Science*, pages 367–382, Berlin, 1998. Springer.

[20] C.N. Potts. An algorithm for the single machine sequencing problem with precedence constraints. *Math. Prog. Stud.*, 13:78–87 (1980)

[21] M. Queyranne and Y. Wang. Single-machine scheduling polyhedra with precedence constraints. *Mathematics of Operations Research*, 16(1):1–20, February 1991.

[22] R. Ravi, A. Agrawal, and P. Klein. Ordering problems approximated: single-processor scheduling and interval graph completion. *ICALP*, Springer Verlag, 751–62 (1991).

[23] W. Smith. Various optimizers for single-stage production. *Naval Res. Logist. Quart.*, 3:59–66 (1956).

[24] J. Sydney. Decomposition algorithms for single-machine sequencing with precedence relations and deferral costs. *Operations Research*, 23(2):283–298 (1975).