On the Existence of Schedules that are Near-Optimal for both Makespan and Total Weighted Completion Time

Cliff Stein* Joel Wein†

Abstract

We give a simple proof that, for any instance of a very general class of scheduling problems, there exists a schedule of makespan at most twice that of the optimal possible *and* of total weighted completion time at most twice that of the optimal possible. We then refine the analysis, yielding variants of this theorem with improved constants, and give some algorithmic consequences of the technique.

Keywords: Deterministic sequencing: single machine and multiple machine

1 Introduction

The theory of scheduling is now over 40 years old, and since the first papers in the mid-1950s numerous scheduling algorithms have been designed to optimize many sorts of optimality criteria in a wide variety of scheduling models; see Lawler, Lenstra, Rinooy Kan and Shmoys [17] for a detailed survey. Two fundamental and much-studied optimality criteria are the makespan (schedule length) and the average weighted completion time, and there has been much work on the design of algorithms that give either optimal solutions or approximately-optimal solutions for both optimality criteria. For the most part, however, efforts to understand each criterion have been disjoint, and there has been relatively little work on understanding their interaction. The major contribution of this paper is to make a general statement about the interaction of these criteria in a wide class of scheduling models; specifically, we show that for many scheduling problems, a schedule exists that is within a small constant factor, simultaneously, with respect to both of these criteria.

We know of almost no results of this sort before our work. The one example of which we are aware is the scheduling of jobs on parallel identical machines: the list-schedule in which jobs are ordered by non-increasing ratio of weight to processing times is known to have makespan at most twice optimal [8] and average weighted completion time at most $(\sqrt{2}+1)/2$ times optimal [15]; if all the weights are equal, this schedule is in fact optimal for average completion time [5]. It was not clear that schedules of high quality with respect to both optimality criteria could exist for more complex models. For example, our work is

*`cliff@cs.dartmouth.edu`. Department of Computer Science, Sudikoff Laboratory, Dartmouth College, Hanover, NH. Research partly supported by NSF Award CCR-9308701, a Walter Burke Research Initiation Award and a Dartmouth College Research Initiation Award. Part of this research was carried out while this author was visiting Stanford University.

†`wein@mem.poly.edu`. Department of Computer Science, Polytechnic University, Brooklyn, NY, 11201. Research partially supported by NSF Research Initiation Award CCR-9211494 and a grant from the New York State Science and Technology Foundation, through its Center for Advanced Technology in Telecommunications.

in sharp contrast to results by Hurkens and Coster on the scheduling of unrelated machines. In that scheduling model each job j must be assigned to some machine i, and requires p_{ij} contiguous time units when processed on machine $i = 1, \ldots, m$. Hurkens and Coster gave instances of this problem for which any schedule with optimal average completion time has makespan $\Omega(\log n)$ times optimal [14].

Our result demonstrates that, for this problem and many others, if one does not require a schedule that is optimal with respect to average weighted completion time, but only near-optimal, then there exists a schedule that is near-optimal with respect to makespan as well.

Our results apply to any scheduling problem that can be characterized as follows. We are given n jobs and m resources; one can think of the resources as machines. With each job j we associate a non-negative weight w_j. A *schedule* is an assignment of jobs to resources over time. We require that valid schedules for the problem satisfy the following two conditions.

1. Truncation at time t: If we take a valid schedule S and remove from it all jobs that complete after time t, the schedule remains a valid schedule for those jobs that remain.

2. Composition: Given two valid schedules S_1 and S_2 for two sets J_1 and J_2 of jobs (where $J_1 \cap J_2$ is potentially nonempty), the composition of S_1 and S_2, obtained by appending S_2 to the end of S_1, and removing from S_2 all jobs that are in $J_1 \cap J_2$, is a valid schedule for $J_1 \cup J_2$.

Note that these conditions are quite general. They apply, for example, to scheduling one machine, parallel identical or unrelated machines, open shops, flow shops, no-wait flow shops, job shops, shops with sets of parallel machines, etc. We may consider models that allow preemption and those that do not, models in which jobs have release date constraints and/or precedence constraints, models in which jobs require service on one machine at a time, a set of machines at one time, a sequence of machines, or numerous other variations. Furthermore, if the model satisfies these constraints, our results apply to *any* set of non-negative weights $w_j, j = 1, ..., n$.

These conditions do not apply, however, to settings in which machines change speed over time or in which jobs may only run at certain times – in both settings composition of valid schedules might not yield a valid schedule. For ease of exposition, in the rest of this note we will make claims about "any" scheduling problem, and mean any problem that satisfies the two conditions above.

Given a reference schedule S, we let C_j^S be the completion time of job j in schedule S. The makespan of schedule S is defined as $C_{\max} = \max_{j=1,\ldots,n} C_j$, and the optimal makespan for a particular instance is denoted by $C_{\max}^{\text{opt}}$. We define the average weighted completion time of a schedule S to be $\frac{1}{n} \sum w_j C_j^S$, whose optimization is equivalent to the optimization of *total* weighted completion time $\sum w_j C_j^S$; we denote the *optimal* average weighted completion time by $\frac{1}{n} \sum w_j C_j^*$.

If a schedule S has $C_{\max}^S \leq \alpha C_{\max}^{\text{opt}}$ and $\sum w_j C_j^S \leq \beta \sum w_j C_j^*$ we call S an (α, β)-schedule. We will call a polynomial-time algorithm that always produces an (α, β)-schedule an (α, β)-algorithm.

Our basic result is a simple proof of the following fact: For any instance, including any set of weights $w_j, j = 1, \ldots, n$ of any scheduling problem, there exists a $(2, 2)$-schedule. The result does not give a polynomial-time algorithm to find such a schedule, but rather shows that, given two schedules that are optimal, respectively, for makespan and average weighted completion time, one can construct from these schedules a $(2,2)$-schedule. We then give a somewhat more complicated argument that establishes the existence of a $(1.88, 1.88)$-schedule for any scheduling problem; in fact, we give a parameterized expression that provides a spectrum of results. We also show that there exist instances of (simple) scheduling problems for which there are no (α,β)-schedules with α,β both simultaneously less than $(\sqrt{5}+1)/2$.

We also give some simple algorithmic corollaries and an application to the traveling salesman problem.

Previous Related Work: The general topic of scheduling so as to optimize more than one criterion has received relatively little attention. The results to date can be broadly grouped into two classes. The first study the optimization of one criterion given a fixed (usually optimal) value for a second criterion; for example, Smith studied the minimization of total completion time on one machine subject to minimal maximum lateness [22], and Shmoys and Tardos gave algorithms for scheduling unrelated machines that optimize average completion time given a fixed makespan [21]. The second class of results focuses on simultaneous optimization; most of the results attempt to characterize a set $\mathcal{S}$ of schedules with corresponding pairs of values for two (or more) criteria that are "Pareto-optimal", which means that no schedules exist that are superior simultaneously in both criteria to any schedule in $\mathcal{S}$. Notable results of this sort are due to Van Wassenhove and Gelders [23], Nelson, Sarin and Daniels [20], Garey, Tarjan and Wilfong [6], McCormick and Pinedo [19], Hoogeveen [10, 11], Hoogeveen and Van de Velde [12]. Until quite recently there was essentially nothing in the way of general algorithmic techniques for bicriteria scheduling, or results about the *quality* of approximation that might be obtained simultaneously.

Recently, Chakrabarti, Phillips, Schulz, Shmoys, Stein & Wein [4] gave a general technique for designing algorithms that simultaneously optimize both makespan and average weighted completion time. They require an algorithm for the *maximum scheduled weight problem:* given a deadline D, schedule a set of jobs of maximum possible weight by time D. From a dual ρ-approximation algorithm for this problem, they construct an algorithm that produces a schedule that has makespan at most 2.89ρ times optimal and average weighted completion time at most 4ρ times optimal; the result also holds with the constants reversed. If our focus is on structural results and not polynomial-time algorithms, we may take $\rho = 1$ and obtain the result that there exist (2.89,4) schedules and (4, 2.89) schedules for any scheduling problem.

In this paper we give a much simpler proof with significantly better constants; in a few isolated cases the technique yields improved algorithms as well.

2 Basic Result

Theorem 1 *For any scheduling problem there exists a* $(2,2)$*-schedule.*

Proof:

Let M be a schedule with optimal makespan $C^M_{\max} = \ell$, and T a schedule with optimal total weighted completion time. We construct from M and T a $(2,2)$-schedule as follows.

1. Remove from T all jobs that complete in T after time ℓ; call this schedule T'.

2. Let M' be the schedule obtained from M by removing from M all jobs that are scheduled in T'; we do not change the scheduling of any job that remains in M'.

3. Form the schedule $N = T'$ followed immediately by M'.

The fact that N is a valid schedule follows from the conditions on our scheduling model: truncation and composition of schedules yields valid schedules. The proof that N is a $(2,2)$-schedule follows from the following two observations. First note that $C_{\max}^N \leq 2\ell = 2C_{\max}^M$. Second, $C_j^N \leq 2C_j^T$, since if a job is scheduled in T', $C_j^N = C_j^T$, and for jobs j scheduled in M' $C_j^N \leq 2\ell$, whereas $C_j^T \geq \ell$. ■

Corollary 2 *For any $\delta > 0$, for any scheduling problem, there exists a $(1+\delta, \frac{1+\delta}{\delta})$ schedule.*

Proof: Construct T' from T by removing all jobs that complete later than $\delta\ell$, $\delta > 0$, and proceed as in the previous proof. The makespan of the resulting schedule is $(1+\delta)\ell$, and $C_j^N \leq \frac{\delta+1}{\delta}C_j^T$, since for jobs scheduled in M' $C_j^N \leq (1+\delta)\ell$, whereas $C_j^T \geq \delta\ell$. ■

Since this simple argument can be applied to schedules that are only approximately optimal (but can be found in polynomial time), we obtain the following algorithmic corollary. We define a ρ-approximation algorithm to be an algorithm that delivers a schedule of value at most ρ times optimal (with respect to one particular optimality criterion).

Corollary 3 *For any scheduling problem, if there exists an α-approximation algorithm for makespan and a β-approximation algorithm for average weighted completion time, there exists an $(\alpha(1+\delta), \beta(\frac{\delta+1}{\delta}))$-algorithm for any $\delta > 0$.*

Proof: Let M be the schedule produced by the approximation-algorithm for makespan, and T the schedule produced by the approximation algorithm for total weighted completion time. Following the construction of Corollary 2 we obtain that $C_{\max}^N \leq (1+\delta)C_{\max}^M \leq \alpha(1+\delta)C_{\max}^{\text{opt}}$, and $\sum_j w_j C_j^N \leq \frac{\delta+1}{\delta}\sum_j w_j C_j^T \leq \beta\frac{\delta+1}{\delta}\sum_j w_j C_j^*$. ■

For some scheduling models this corollary yields better bicriteria algorithms than what can be achieved by the technique of Chakrabarti et. al. [4]; we discuss these after we have presented the improved analysis in the next section.

We conclude our collection of corollaries with an application to the traveling salesman problem. The traveling salesman problem can be stated as follows: We are given a set of n cities, and a distance matrix d_{ij}, $i = 1, \ldots, n$, and $j = 1, \ldots, n$, with $d_{ij} \geq 0$; we will assume that the d_{ij} satisfy the triangle inequality; namely, for any i, j, k, $d_{ij} \leq d_{ik} + d_{kj}$. A tour is defined as a permutation of the n cities that defines the order in which they are to be traversed. The classical Traveling Salesman Problem (TSP) is to produce a tour of minimum length [16]. A variant is known as the Traveling Repairman Problem (TRP) [1, 2, 7]. In this variant we have a designated start vertex v, and define c_i to be the distance in the tour from vertex v to vertex i, and associated with vertex i a nonnegative weight w_i; the goal is to find a tour that minimizes $\sum_i w_i c_i$. It is clear that the traveling salesman problem is analogous to a makespan problem, and the traveling repairman problem is analogous to an average weighted completion time problem.

Theorem 4 *For any instance of the traveling salesman problem with triangle inequality, there exists a tour of length within a factor of 2 of the optimal possible and that simultaneously has $\sum_i w_i c_i$ at most twice that of the optimal traveling repairman tour.*

Proof: Let T_1 be a tour, of length t_1, that is optimal for the TSP, and T_2 a tour that is optimal for the TRP. Take T_2 and remove from it all vertices that are farther than t_1 from the start vertex in the tour; denote by T_2' the path that remains from T_2 and let v' be the last vertex in T_2' (in other words, the last vertex on T_2 with $c_i \leq t_1$).

Consider v', v and all vertices that are not in T_2', and construct the tour on these vertices that is induced by T_1; by the triangle inequality, the length of this induced tour is no more than t_1. Use this ordering to complete T_2' to be a complete tour on all n vertices. It is clear that the length of T_2' is at most $2t_1$, and, in the same spirit as the previous arguments in this section, that the c_i value for vertex i is at most doubled. ■

3 Improved Analysis

In this section we give a more detailed analysis that improves upon the results of the previous section. The basic intuition is that, in the construction of the previous section, it is unlikely that *all* of the weight was doubled, and that therefore the average weighted completion time of the new schedule N may be less than twice that of T. By considering more carefully the distribution of weight in T, and choosing, depending on that distribution, between several possible points at which to truncate T, we obtain improved bounds. We continue to use the notation from the previous section; namely M is the schedule that is optimal for the makespan, T is optimal for average weighted completion time, and N is the new schedule constructed from T and M. Our construction remains simple: choose a point at which to truncate T and then append M, removing from M those jobs that were already scheduled in T.

Theorem 5 *Let g be a constant such that $1 < g \leq 2$, and let γ be a constant such that $0 \leq \gamma < 1$. If there exist constants α, β and δ such that the following four conditions hold:*

$$0 < \alpha < \beta < \gamma \leq 1 \tag{1}$$

$$0 < \delta < 1 \tag{2}$$

$$1 + \frac{(1-\delta)(1-\gamma(g-1))}{\beta} + (1+\beta-\gamma)(g-1) \leq g \tag{3}$$

$$1 + \frac{\delta(1-\gamma(g-1))}{\alpha} + \frac{(1+\alpha-\beta)(1-\delta)(1-\gamma(g-1))}{\beta} + (1+\alpha-\gamma)(g-1) \leq g \tag{4}$$

then there is a $(1+\gamma, g)$ schedule for any scheduling problem.

Proof: Consider T, the schedule of optimal average weighted completion time. We divide time into four intervals: $I_0 = [0, \alpha\ell], I_1 = [\alpha\ell, \beta\ell], I_2 = [\beta\ell, \gamma\ell], I_3 = [\gamma\ell, \infty]$, and let $W_i = \sum_{j: C_j^T \in I_i} w_j C_j$ and define $W^* = \sum_j w_j C_j^T = \sum_j w_j C_j^*$. We will consider truncating T at times $\alpha\ell$, $\beta\ell$ and $\gamma\ell$. If we truncate T at time t, then jobs that complete in T before time t do not have their completion times affected. Jobs that complete in T after time t have their average completion time increased by at most the ratio of the new schedule's makespan divided by the left endpoint of the interval in which they were scheduled in T.

With this in mind, we can give bounds on the average weighted completion time when we truncate at point $x\ell$, which we will denote C^x_{avg}. If we truncate at $\gamma\ell$, then

$$
\begin{aligned}
C^\gamma_{\text{avg}} &\leq W_0 + W_1 + W_2 + \frac{1+\gamma}{\gamma} W_3 \\
&= W_0 + W_1 + W_2 + W_3 + \frac{1}{\gamma} W_3 \\
&= W^* + \frac{1}{\gamma} W_3. \qquad (5)
\end{aligned}
$$

If we truncate at $\beta\ell$, then

$$
\begin{aligned}
C^\beta_{\text{avg}} &\leq W_0 + W_1 + \frac{1+\beta}{\beta} W_2 + \frac{1+\beta}{\gamma} W_3 \\
&= W^* + \frac{1}{\beta} W_2 + \frac{1+\beta-\gamma}{\gamma} W_3. \qquad (6)
\end{aligned}
$$

If we truncate at $\alpha\ell$, then

$$
\begin{aligned}
C^\alpha_{\text{avg}} &\leq W_0 + \frac{1+\alpha}{\alpha} W_1 + \frac{1+\alpha}{\beta} W_2 + \frac{1+\alpha}{\gamma} W_3 \\
&= W^* + \frac{1}{\alpha} W_1 + \frac{1+\alpha-\beta}{\beta} W_2 + \frac{1+\alpha-\gamma}{\gamma} W_3. \qquad (7)
\end{aligned}
$$

We now consider three cases for where the weight is distributed. In the first case not too much weight completes in I_3, and hence truncation at γL will not increase the completion time of too much of the weight. If W_3 is large, we truncate at either $\beta\ell$ or $\alpha\ell$, depending on the relative values of W_2 and W_1. The key observation is that the jobs that complete soon after the truncation point in T have their completion times increased the most in the new schedule N; we therefore wish to keep this quantity small.

In the following arguments, we assume that $W_0 = 0$. This only worsens our bounds, as the jobs in I_0 never get moved later in the schedule and hence this is the "best" place to have jobs finish.

More formally:

Case 1: $W_3 \leq \gamma(g-1)W^*$. In this case we truncate at $\gamma\ell$; our bound on C^γ_{avg} from (5) yields that $C^\gamma_{\text{avg}} \leq W^* + \frac{1}{\gamma}W_3 \leq W^* + (g-1)W^* \leq gW^*$.

Case 2: $W_3 > \gamma(g-1)W^*, W_1 \geq \delta(1-\gamma(g-1))W^*$. In this case, I_2 has relatively little weight, and so we truncate at $\beta\ell$. Note that the upper bound on C^β_{avg} is a linear function of W_2 and W_3, in which W_2 has the larger coefficient. Thus we maximize the upper bound on C^β_{avg} by first maximizing W_2, and then maximizing W_3, subject to the constraints of this case.

To maximize W_2, we observe that $W_2 = W^* - W_1 - W_3$ and minimize W_1 and W_3 subject to the lower bounds on these quantities that define this case. This gives us that $W_2 = (1-\delta)(1-\gamma(g-1))W^*$. Similarly, we get $W_3 = \gamma(g-1)W^*$. Utilizing our upper bound for C^β_{avg} from (6) gives that

$$
C^\beta_{\text{avg}} \leq W^* \left(1 + \frac{(1-\delta)(1-\gamma(g-1))}{\beta} + (1+\beta-\gamma)(g-1)\right) \leq gW^* \qquad (8)
$$

where the last inequality follows condition (3) of the theorem.

Case 3: $W_3 > \gamma(g-1)W^*, W_1 < \delta(1-\gamma(g-1))W^*$. In this case, I_1 does not have much weight and so we truncate at $\alpha\ell$. Note that our upper bound on C^{α}_{avg} from (6) is a linear function of W_1, W_2 and W_3, and that W_1 has the largest coefficient, followed by W_2, and then W_3. Thus C^{α}_{avg} is maximized by first maximizing W_1, and then maximizing W_2, and then maximizing W_3 subject to the constraints of this case. This gives us that $W_1 = \delta(1-\gamma(g-1))W^*$, $W_2 = (1-\delta)(1-\gamma(g-1))W^*$ and $W_3 = \gamma(g-1)W^*$. Utilizing our upper bound for for C^{α}_{avg} above (7) gives that

$$C^{\alpha}_{\text{avg}} \leq W^* \left(1 + \frac{\delta(1-\gamma(g-1))}{\alpha} + \frac{(1+\alpha-\beta)(1-\delta)(1-\gamma(g-1))}{\beta} + (1+\alpha-\gamma)(g-1)\right) \quad (9)$$

which is, by the conditions of the theorem, at most gW^*.

We do not know, a priori, which case will apply, but certainly the maximum of these three upper bounds is an upper bound on on the average weighted completion time. The conditions of the lemma are clearly sufficient conditions for the maximum of the three upper bounds to be gW^*.

The schedules have as their makespan $1+\gamma$, $1+\beta$ and $1+\alpha$ respectively. Hence the makespan is at most $1+\gamma$. ■

One can derive many different tradeoffs between makespan and average weighted completion time using this theorem; we give three examples here.

Corollary 6 *For any instance of any scheduling problem there exists a* $(2, 1.735)$ *schedule, a* $(1.785, 2)$ *schedule, and a* $(1.88, 1.88)$ *schedule.*

Proof: For the first set $\alpha = .43$, $\beta = .71$, and $\delta = .43$. For the second set $\alpha = .35$, $\beta = .58$, and $\delta = .45$. For the third set $\alpha = .46$, $\beta = .69$, and $\delta = .49$. ■

The improved analysis can be used to improve the corollaries of the previous section as well.

Corollary 7 *Let g, γ satisfy the conditions of Theorem 5.*

- *For any scheduling problem, if there exists an α-approximation algorithm for makespan and a β-approximation algorithm for average weighted completion time, there exists a $(\alpha(1+\gamma), \beta g)$ schedule for any scheduling problem.*

- *For any instance of the TSP with triangle inequality, there exists a tour of length within a factor of $1+\gamma$ of the optimal possible and that simultaneously has $\sum_i w_i c_i$ at most a factor of g of the optimal TRP tour.*

For some scheduling models this corollary yields better bicriteria algorithms than what can be achieved by the technique of Chakrabarti et. al. [4]; To begin, consider the special case of $w_j = 1$ for all j. For the problem of scheduling jobs on parallel machines of different speeds, Hochbaum & Shmoys have given a polynomial-approximation scheme to minimize makespan [9], and Horn [13] and Bruno, Coffman and Sethi [3] gave a polynomial-time algorithm to minimize average completion time; taking $\delta = 1$ we obtain a $(1.88+\epsilon, 1.88)$-algorithm in this model. For the more general problem of scheduling jobs on *unrelated* parallel machines there is a 2-approximation algorithm for makespan [18] and a polynomial-time algorithm for average completion time [13, 3]; thus there exists a $(3.76, 1.88)$-algorithm. Finally, for the scheduling of jobs, now with general weights, on parallel identical machines, there is a $(\frac{\sqrt{2}+1}{2})$-approximation algorithm for average weighted completion time [15]; combined with the polynomial-approximation scheme of Hochbaum and Shmoys we obtain a $(1.88+\epsilon, 2.27)$-algorithm. All of these results are clearly better than the best possible via the technique of Chakrabarti et. al. [4].

As a final consequence of our results we note that, since the method in which we construct a (α, β)-schedule bounds not only the total weighted completion time by a factor of β times optimal, but in addition bounds the completion time of *each job* by a factor of β times its completion in a schedule that is optimal for average weighted completion time, our results have consequences for minsum criteria other than $\sum w_j C_j$, such as $\sum w_j C_j^2$.

Corollary 8 *Let g, γ satisfy the conditions of Theorem 5, and let $\theta \geq 1$. Then there is, for any scheduling problem, a schedule S that simultaneously has makespan of at most a factor of $(1+\gamma)$ times the optimal possible makespan, and has $\sum w_j (C_j^S)^\theta$ at most a factor of g^θ times that of the optimal schedule for this criterion.*

Proof: We follow the same construction as in the previous results, replacing the schedule that is optimal for $\sum w_j C_j$ with one that is optimal for $\sum w_j (C_j^S)^\theta$. ■

4 Lower Bound

Theorem 9 *There exist scheduling instances for which there is no (α, β)-schedule with both α and β simultaneously less than $(\sqrt{5}+1)/2$.*

Consider the problem of the nonpreemptive scheduling of jobs with release dates on one machine, and consider the following simple instance: a job of size x and weight 1 released at time 0, and a job of size 1 and weight $w >> 1$ released at αx, $0 \leq \alpha < 1$ (and therefore is not available for processing before time αx). The optimal makespan for this instance is $x+1$, and the optimal total weighted completion time is $w(\alpha x+1)+((\alpha+1)x+1)$. These bounds, however, can not be achieved simultaneously. There are two reasonable schedules for this instance. In the first, the small job is scheduled after the large job, giving a makespan of $x+1$ and a total weighted completion time of $x+w(x+1)$. In the second schedule the small job is scheduled at its release time, giving a makespan of $\alpha x + x + 1$ and a total weighted completion time of $w(\alpha x+1)+(\alpha x+x+1)$. The first schedule is thus, for sufficiently large w, no better than a $(1, \frac{1}{\alpha})$-schedule and the second no better than a $((1+\alpha), 1)$-schedule. Choosing $\alpha = (\sqrt{5}-1)/2$ gives our theorem. ■

5 Acknowledgments

We thank Soumen Chakrabarti, Leslie Hall and David Shmoys for useful discussions.

References

[1] F. Afrati, S. Cosmadakis, C. Papadimitriou, G. Papageorgiou, and N. Papakostantinou. The complexity of the travelling repairman problem. *Informatique Theoretique et Applications*, pages 79–87, 1986.

[2] A. Blum, P. Chalasani, D. Coppersmith, B. Pulleyblank, P. Raghavan, and M. Sudan. The minimum latency problem. In *Proceedings of the 26th Annual ACM Symposium on Theory of Computing*, pages 163–172, May 1994.

[3] J.L. Bruno, E.G. Coffman, and R. Sethi. Scheduling independent tasks to reduce mean finishing time. *Communications of the ACM*, 17:382–387, 1974.

[4] S. Chakrabarti, C. A. Phillips, A. S. Schulz, D. B. Shmoys, C. Stein, and J. Wein. Improved scheduling algorithms for minsum criteria. In F. Meyer auf der Heide and B. Monien, editors, *Automata, Languages and Programming*, number 1099 in Lecture Notes in Computer Science. Springer, Berlin, 1996. Proceedings of the 23rd International Colloquium (ICALP'96).

[5] R.W. Conway, W.L. Maxwell, and L.W. Miller. *Theory of Scheduling.* Addison-Wesley, 1967.

[6] M. R. Garey, R.E. Tarjan, and G. T. Wilfong. One-processor scheduling with symmetric earliness and tardiness penalties. *Mathematics of Operations Research*, 13:330–348, 1988.

[7] M. Goemans and J. Kleinberg. An improved approximation ratio for the minimum latency problem. In *Proceedings of the 7th ACM-SIAM Symposium on Discrete Algorithms*, pages 152–157, 1996.

[8] R.L. Graham. Bounds for certain multiprocessor anomalies. *Bell System Technical Journal*, 45:1563–1581, 1966.

[9] D.S. Hochbaum and D.B. Shmoys. A polynomial approximation scheme for machine scheduling on uniform processors: using the dual approximation approach. *SIAM Journal on Computing*, 17:539–551, 1988.

[10] J. A. Hoogeveen. Minimizing maximum promptness and maximum lateness on a single machine. *Mathematics of Operations Research*, 21:100–114, 1996.

[11] J. A. Hoogeveen. Single-machine scheduling to minimize a function of two or three maximum cost criteria. *Journal of Algorithms*, 21(2):415–433, 1996.

[12] J. A. Hoogeveen and S.L. van de Velde. Minimizing total completion time and maximum cost simulatneously is solvable in polynomial time. *Operations Research Letters*, 17:205–208, 1995.

[13] W. Horn. Minimizing average flow time with parallel machines. *Operations Research*, 21:846–847, 1973.

[14] C.A.J. Hurkens and M.J. Coster. On the makespan of a schedule minimizing total completion time for unrelated parallel machines. Unpublished manuscript, 1996.

[15] T. Kawaguchi and S. Kyan. Worst case bound of an LRF schedule for the mean weighted flow-time problem. *SIAM Journal on Computing*, 15:1119–1129, 1986.

[16] E.L. Lawler, J.K. Lenstra, A.H.G. Rinooy Kan, and D.B. Shmoys, editors. *The Traveling Salesman Problem.* John Wiley and Sons, 1985.

[17] E.L. Lawler, J.K. Lenstra, A.H.G. Rinooy Kan, and D.B. Shmoys. Sequencing and scheduling: Algorithms and complexity. In S.C. Graves, A.H.G. Rinnooy Kan, and P.H. Zipkin, editors, *Handbooks in Operations Research and Management Science, Vol 4., Logistics of Production and Inventory*, pages 445–522. North-Holland, 1993.

[18] J.K. Lenstra, D.B. Shmoys, and É. Tardos. Approximation algorithms for scheduling unrelated parallel machines. *Mathematical Programming*, 46:259–271, 1990.

[19] S.T. McCormick and M.L. Pinedo. Scheduling n independent jobs on m uniform machines with both flow time and makespan objectives: A parametric approach. *ORSA Journal of Computing*, 7:63–77, 1992.

[20] R.T. Nelson, R.K. Sarin, and R.L. Daniels. Scheduling with multiple performance measures: the one-machine case. *Management Science*, 32:464–479, 1986.

[21] D. B. Shmoys and E. Tardos. An approximation algorithm for the generalized assignment problem. *Mathematical Programming A*, 62:461–474, 1993.

[22] W.E. Smith. Various optimizers for single-stage production. *Naval Research Logistics Quarterly*, 3:59–66, 1956.

[23] L.N. Van Wassenhove and F. Gelders. Solving a bicriterion scheduling problem. *European Journal of Operations Research*, 4:42–48, 1980.