The Constrained Minimum Weighted Sum of Job Completion Times Problem

Asaf Levin[1] and Gerhard J. Woeginger[2,3]

[1] Faculty of Industrial Engineering and Management, Technion
Haifa 32000, Israel
levinas@tx.technion.ac.il
[2] Department of Mathematics, University of Twente, P.O. Box 217
7500 AE Enschede, The Netherlands
g.j.woeginger@math.utwente.nl
[3] Department of Mathematics and Computer Science, TU Eindhoven
P.O. Box 513, 5600 MB Eindhoven, The Netherlands
gwoegi@win.tue.nl

Abstract. We consider the problem of minimizing the weighted sum of job completion times on a single machine (subject to certain job weights) with an additional side constraint on the weighted sum of job completion times (with respect to different job weights). This problem is NP-hard, and we provide a polynomial time approximation scheme for this problem. Our method is based on Lagrangian relaxation mixed with carefully guessing the positions of certain jobs in the schedule.

Keywords: scheduling; bicriteria optimization; approximation scheme.

1 Introduction

Consider a set of n jobs $1, 2, \ldots, n$ where each job j has a positive integer processing time p_j, and two positive integer weights w_j and u_j. These jobs have to be scheduled on a single machine. A *schedule* essentially corresponds to a permutation $\pi \in S_n$ of the jobs, where $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ lists the jobs in the schedule beginning with the first one and ending with the last one. A schedule π induces for every job j a *completion time* C_j^π; if the schedule π is clear from the context, we will suppress the superscript π and simply write C_j. In this paper, we study the following bicriteria version of the problem of minimizing the sum of weighted job completion times:

$$\begin{aligned} &\text{minimize} && \textstyle\sum_{j=1}^{n} w_j C_j^\pi \\ &\text{subject to} && \textstyle\sum_{j=1}^{n} u_j C_j^\pi \;\leq\; U \\ & && \pi \in S_n \end{aligned} \tag{1}$$

Here the integer U is a budget bound that is specified as part of the input. This optimization problem is called the CONSTRAINED MINIMUM WEIGHTED SUM OF JOB COMPLETION TIMES PROBLEM ON A SINGLE MACHINE (CWSCT). It arguably belongs to the most simple non-trivial bicriteria scheduling problems.

D. Bienstock and G. Nemhauser (Eds.): IPCO 2004, LNCS 3064, pp. 298–307, 2004.

Our Results. In Section 2 we give a straightforward reduction from the partition problem that shows that CWSCT is NP-hard. As our main result, we then construct in Section 3 a polynomial time approximation scheme (PTAS) for CWSCT: A *ρ-approximation algorithm* is a polynomial time algorithm that returns a near-optimal feasible solution with cost at most a factor ρ above the optimal cost. A *polynomial time approximation scheme* (PTAS) is a family of $(1+\varepsilon)$-approximation algorithms over all $\varepsilon > 0$. Our method is based on Lagrangian relaxation, on adjacency relations between various optimal permutations, and on a number of guessing steps that guess the positions of certain jobs in an optimal schedule.

Known Results. The literature contains a number of results on bicriteria scheduling problems: McCormick & Pinedo [6] consider the problem of scheduling n jobs on m uniform machines with job preemptions. They present a polynomial time algorithm that generates the entire trade-off curve of schedules which are Pareto-optimal for the makespan and the flowtime objectives. Shmoys & Tardos [9] consider a scheduling problem with n jobs and m unrelated machines. Assigning job j to machine i causes a processing time p_{ij} on this machine and also a global cost c_{ij}. For a given (hard) budget constraint C on the total cost and for a given bound T on the makespan, the algorithm in [9] either proves that there is no schedule of cost at most C and makespan at most T, or otherwise it provides a schedule with cost at most C and makespan at most $2T$. Hence, this yields a kind of 2-approximation algorithm for this budget problem. The results in [9] improve on earlier results by Lin & Vitter [5] on the same problem. Stein & Wein [11] consider a fairly general family of bicriteria scheduling problems. They show that all problems in their family possess schedules that simultaneously approximate the optimal makespan within a factor of 1.88 and the optimal sum of weighted job completion times within a factor of 1.88. Aslam, Rasala, Stein & Young [1] and Rasala, Stein, Torng & Uthaisombut [7] provide a variety of other results in this direction.

There is a rich literature on bicriteria problems in graph theory; see Ravi [8] for a (restricted) survey. Here we only want to mention a result by Goemans & Ravi [3] that yields a PTAS for the constrained minimum spanning tree problem. The approach in [3] is based on a Lagrangian relaxation approach, and it heavily exploits the adjacency structure of the underlying matroid polytope. When we started to work on CWSCT, we were hoping to be able to apply the machinery of [3] more or less directly. However, the lack of matroid properties in CWSCT seems to make this impossible; we worked around this by introducing a number of guessing steps that provide us with the necessary combinatorial structures.

2 NP-hardness

In this section we prove NP-hardness of CWSCT via a reduction from the NP-hard PARTITION problem (see Garey & Johnson [2]). An instance of PARTITION consists of n positive integers $a_1, a_2, \ldots, a_n$ such that $\sum_{i=1}^{n} a_i = 2A$ where A is

an integer. The problem is to decide whether there exists an index set S such that $\sum_{i\in S} a_i = \sum_{i\notin S} a_i = A$.

From an instance of PARTITION we define an instance of CWSCT as follows: Let $M = 4A^2 + 1$. There are $n+1$ jobs $1, 2, \ldots, n, n+1$. The jobs j with $1 \leq j \leq n$ have $p_j = w_j = a_j$ and $u_j = 0$, whereas the last job has $p_{n+1} = M$, $u_{n+1} = 1$, and $w_{n+1} = 0$. Finally, the budget U equals $M + A$. We claim that the PARTITION instance has answer YES, if and only if the CWSCT instance has a schedule with objective value at most $4A^2 + MA$.

First assume that the PARTITION instance has answer YES, and that there is an index set S with $\sum_{i\in S} a_i = A$. We construct a schedule π that first runs the jobs in S (in arbitrary order), then runs job $n+1$, and finally runs the remaining jobs (in arbitrary order). In schedule π job $n+1$ completes exactly at time $M + \sum_{i\in S} a_i = M + A = U$. Since all other jobs have zero u-weight, the schedule π indeed satisfies the budget constraint, and hence is feasible. The total w-weight of the jobs that are processed after job $n+1$ exactly equals A. Therefore, the objective value of π is

$$\sum_{j=1}^{n+1} w_j C_j \;\leq\; \sum_{j=1}^{n+1} \sum_{i:\pi_i \geq \pi_j} p_j w_i \;\leq\; (\sum_{i=1}^{n} p_j)(\sum_{j=1}^{n} w_i) + p_{n+1} \sum_{i\notin S} w_i$$

$$= \; 4A^2 + MA.$$

Next, assume that the constructed CWSCT instance has a schedule with objective value at most $4A^2 + MA$. We denote by S the set of jobs in π that are processed before job $n+1$ starts. We will show that $\sum_{i\in S} a_i = A$, and hence the PARTITION instance has answer YES. Since $C_{n+1} = \sum_j u_j C_j \leq U = M + A$ and since $C_{n+1} = p_{n+1} + \sum_{i\in S} p_i = M + \sum_{i\in S} p_i$, we conclude $\sum_{i\in S} p_i \leq A$. Therefore,

$$\sum_{i\in S} a_i \leq A. \tag{2}$$

Suppose for the sake of contradiction that $\sum_{i\notin S} a_i > A$. Then the integrality of the numbers a_i yields $\sum_{i\notin S} w_i = \sum_{i\notin S} a_i \geq A + 1$. The jobs in $\{1, 2, \ldots, n\} \setminus S$ are all processed after job $n+1$, and hence have completion times of at least $p_{n+1} = M$. Then, the total cost of the solution is at least $M \sum_{i\notin S} w_i \geq M(A + 1) > MA + 4A^2$, where the last inequality follows because $M > 4A^2$. But now the objective value of π would be greater than $4A^2 + MA$, the desired contradiction. Therefore, $\sum_{i\notin S} a_i \leq A$ holds, which together with (2) yields $\sum_{i\in S} a_i = A$. Hence, the PARTITION instance has answer YES.

Summarizing, this yields the following theorem.

Theorem 1. *Problem CWSCT is NP-hard in the ordinary sense.*

3 The Polynomial Time Approximation Scheme

Let $\varepsilon > 0$ be the desired precision of approximation (a fixed constant that is not part of the input); to simplify the presentation we will assume that $1/\varepsilon$ is integer.

Consider an instance I of CWSCT. Let $W = \sum_{j=1}^{n} w_j$ denote the total w-weight of the jobs, and let $P = \sum_{j=1}^{n} p_j$ denote their total processing time. We assume without loss of generality that instance I has at least one feasible schedule in (1); this can be checked in polynomial time through Smith's algorithm [10]. Finally, we fix an optimal schedule π^* for instance I relative to which we will do our analysis.

The approximation scheme goes through a number of *guessing* steps that guess certain pieces of information about the structure of the optimal schedule. All our guesses can be emulated in polynomial time either by means of a binary search or by means of total enumeration.

In our first guessing step, we guess the optimal objective value $\mathtt{OPT}$. Since $\mathtt{OPT}$ lies somewhere between 0 and WP, we can approximate it with arbitrary precision through a binary search: If our current guess is too small, the whole procedure will fail, and we know that we have to use a larger guess. If our current guess is too large, the procedure yields an approximate schedule and we may move on to a smaller guess. Hence, the guesses eventually converge to $\mathtt{OPT}$, and we terminate the binary search as soon as the induced lower and upper bounds on $\mathtt{OPT}$ are only a factor $1 + \varepsilon$ away from each other.

Based on $\mathtt{OPT}$, we define the threshold parameter τ:

$$\tau \;=\; \varepsilon\,\mathtt{OPT}. \tag{3}$$

A pair of jobs (i, j) is a *critical* pair if $w_i p_j - w_j p_i \geq \tau$ holds. A critical pair (i, j) is called an *active* critical pair in π if $\pi_j < \pi_i$. A critical pair (i, j) is called an *in-active* critical pair in π if $\pi_j > \pi_i$.

Lemma 1. *In any optimal schedule, there are at most* $1/\varepsilon$ *active critical pairs.*

Proof. Consider an optimal schedule, and assume that some job i participates in exactly k active critical pairs $(i, j_1), (i, j_2), \ldots, (i, j_k)$. Then the contribution of job i to the objective value is at least

$$w_i C_i \;\geq\; w_i \sum_{l=1}^{k} p_{j_l} \;\geq\; \sum_{l=1}^{k} (w_i p_{j_l} - w_{j_l} p_i) \;\geq\; k\,\varepsilon\,\mathtt{OPT}.$$

Therefore, there are at most $1/\varepsilon$ active critical pairs. □

As an immediate consequence of Lemma 1, any optimal schedule has at most $2/\varepsilon$ jobs that participate in some active critical pair. In our second guessing step, we guess these at most $2/\varepsilon$ jobs together with their relative ordering in the optimal schedule π^*. This guessing step amounts to enumerate $O(n^{2/\varepsilon})$ subsets and $(2/\varepsilon)!$ total orderings for each of these subsets, that is, to check a polynomial number of possibilities. The guessed ordering of the guessed jobs yields a chain-like precedence constraint structure $\mathtt{Chain}$ on the job set.

Now we introduce our first version of the Lagrangian relaxation of (1). For a real parameter $\lambda \geq 0$, we define Lagrangian job weights $\ell_j^\lambda = w_j + \lambda u_j$ where $1 \leq j \leq n$. Consider the following optimization problem $P(\lambda)$:

$$P(\lambda) = \min \sum_{j=1}^n \ell_j^\lambda C_j^\pi - \lambda U = \sum_{j=1}^n w_j C_j^\pi + \lambda \left(\sum_{j=1}^n u_j C_j^\pi - U\right) \quad \text{s.t.} \quad \pi \text{ satisfies } \texttt{Chain} \tag{4}$$

For every $\lambda \geq 0$ the value $P(\lambda)$ is a lower bound on OPT, since the optimal permutation π^* for (1) by definition satisfies the precedence constraints Chain and also the budget constraint $\sum_j u_j C_j \leq U$.

Moreover, for every $\lambda \geq 0$, problem $P(\lambda)$ can be solved in polynomial time: It is the scheduling problem of minimizing total weighted job completion time on a single machine under a single chain precedence constraint. In Section 4 we explain how Horn's algorithm [4] solves this problem by grouping the jobs in the chain Chain into a number of bundles. Note that some of these bundles may consist of only one job. For every λ with weights ℓ_j^λ, we denote the resulting set of bundles by $B_\lambda = \{J_1, J_2, \ldots\}$. For every bundle J, we denote by p_J the total processing time of all the jobs in J and by w_J their total w-weight. We define the set D_λ of *dangerous* jobs as:

$$D_\lambda = \{i \notin B_\lambda : \exists J \in B_\lambda \text{ with } |p_J w_i - p_i w_J| \geq \tau\}. \tag{5}$$

Note that a dangerous job is never a job bundle, and also is not part of a bundle. A dangerous job $i \in D_\lambda$ becomes a *misplaced* dangerous job, if there exists a job bundle $J \in B_\lambda$ such that either $w_i p_J - w_J p_i \geq \tau$ and i is scheduled before J in the optimal schedule π^*, or $w_J p_i - w_i p_J \geq \tau$ and J is scheduled before i in the optimal schedule π^*.

Lemma 2. *With respect to any real $\lambda \geq 0$, there are at most $1/\varepsilon$ misplaced dangerous jobs in the optimal schedule π^*.*

Proof. Assume that job i is a misplaced dangerous job with respect to $J_1, J_2, \ldots, J_k \in B_\lambda$, and that i is scheduled after all these jobs J_j. Then

$$w_i C_i \;\geq\; \sum_{j=1}^k w_i p_{J_j} \;\geq\; \sum_{j=1}^k (w_i p_{J_j} - p_{J_j} w_i) \;\geq\; k\,\tau \;\geq\; k\,\varepsilon\,\texttt{OPT}.$$

For $J \in B_\lambda$, let $i_1, i_2, \ldots, i_k$ be the set of misplaced dangerous jobs with respect to J, such that i_l is scheduled before J for all indices l. Then,

$$\sum_{j \in J} w_j C_j \;\geq\; \sum_{j \in J} \sum_{l=1}^k w_j p_{i_l} \;=\; \sum_{l=1}^k w_J p_{i_l} \;\geq\; \sum_{l=1}^k (w_J p_{i_l} - p_J w_{i_l})$$

$$\geq\; k\,\tau \;\geq\; k\,\varepsilon\,\texttt{OPT}.$$

Therefore, there are at most $1/\varepsilon$ misplaced dangerous jobs. □

Our next goal is to find a maximizer λ^* for (4), to find a corresponding schedule and corresponding bundles, and to be in full control of the resulting misplaced dangerous jobs. This is accomplished through our third guessing step: We guess the set of misplaced dangerous jobs, and for each such misplaced dangerous job we guess its exact position within `Chain`. The resulting new precedence constraints are denoted by `Chain'`. Note that this guessing step amounts to enumerate $O(n^{1/\varepsilon})$ subsets and $O((2/\varepsilon)^{1/\varepsilon})$ possibilities for the positions; all in all that is a polynomial amount of work. We arrive at our second version $P'(\lambda)$ of the Lagrangian relaxation:

$$\begin{aligned} P'(\lambda) = \min & \textstyle\sum_{j=1}^{n} \ell_j^{\lambda} C_j^{\pi} - \lambda U = \sum_{j=1}^{n} w_j C_j^{\pi} + \lambda \left(\sum_{j=1}^{n} u_j C_j^{\pi} - U\right) \\ \text{s.t.} \quad & \pi \text{ satisfies } \texttt{Chain}' \end{aligned} \tag{6}$$

For the rest of this section, we will keep the old notation of job bundles exactly as introduced for $P(\lambda)$, and we will not adapt it to the second relaxation $P'(\lambda)$. Note that for all $\lambda \geq 0$ the value $P'(\lambda)$ is (again) a lower bound on `OPT`, since the optimal permutation π^* satisfies the precedence constraints `Chain'` and the budget constraint $\sum_j u_j C_j \leq U$.

Below we show how to determine a a maximizer λ^* for (6) in polynomial time. If the corresponding bundles and misplaced difficult jobs collide with the data of our third guessing step, we simply cut off this guessing branch and move on to the next guess.

Lemma 3. *We may assume without loss of generality that for $\lambda = 0$ the corresponding optimal solution to $P'(\lambda)$ does not satisfy the budget constraint $\sum_{j=1}^{n} u_j C_j \leq U$.*

For $\lambda = 2WP$ the corresponding optimal solution to $P'(\lambda)$ does satisfy the budget constraint $\sum_{j=1}^{n} u_j C_j \leq U$.

Proof. First we assume $\lambda = 0$. Then, the objective function in (6) becomes $\sum_{j=1}^{n} w_j C_j$, that is, it becomes the objective function of (1). Since furthermore the optimal permutation π^* for (1) is feasible for (6), we have $P'(0) \leq$ `OPT`. In case the optimal schedule for (6) satisfies the budget constraint in (1), it is feasible for (1) and hence optimal for (1). In this case there is nothing left to do.

Next we assume $\lambda = 2WP$. If a solution π for (6) does not satisfy the budget constraint, then $\sum_{j=1}^{n} u_j C_j^{\pi} \geq U+1$ and its objective value is at least $\lambda = 2WP$. If a solution π for (6) satisfies the budget constraint (as for instance the solution π^* does), then its objective value is at most WP. Hence, in this case the optimal solution for (6) satifies the budget constraint. □

For technical reasons, we now replace the budget bound U by the infinitesimally larger bound $U+\gamma$. This does not change the combinatorics of the problem, since all the job weights, job processing times, and job completion times are integers. However, it helps us to get rid of certain degeneracies. Our next step is to determine in polynomial time a maximizer λ^* for the second Lagrangian relaxation $P'(\lambda^*)$. This can be done by a straightforward binary search that is based on the following four observations:

- The cost of every permutation π as a function of λ is a linear function. This function crosses the line $\lambda = 0$ in some integer between 0 and WP, and crosses the line $\lambda = 2WP$ in some integer between $-2WPU$ and $2WP^2 \sum_j u_j$. Since we are using the modified budget bound $U + \gamma$, and since all the job data are integers, this linear function can not have slope zero.
- The cost of the optimal solution of (6) as a function of λ is the minimum of a set of linear functions (one linear function for each permutation). Therefore, this cost is a piecewise linear, concave function in λ.
- The (absolute value of the) difference between the slopes of two non-parallel linear functions is at least $1/(2WP)$ and at most $2WP$.
- The distance between two adjacent break-points on the lower envelope of these functions is at least $1/(2WP)$.

Together with the maximizer λ^* we also compute two optimal solutions α and β for $P'(\lambda^*)$, such that the first solution α satisfies the budget constraint $\sum_{j=1}^{n} u_j C_j^{\alpha} \leq U$, whereas the second solution β violates it with $\sum_{j=1}^{n} u_j C_j^{\beta} > U$. The existence of two such solutions can be established by standard observations: The optimal permutations for $\lambda = \lambda^* - \delta$ and $\lambda = \lambda^* + \delta$ for some arbitrarily small $\delta > 0$ are also optimal for λ^*. If all optimal solutions for λ^* would violate the budget constraint, then $P'(\lambda^* + \delta) > P'(\lambda^*)$ and λ^* would not maximize (6). Similarly, if all optimal solutions for λ^* would satisfy the budget constraint, then they would satisfy it with strict inequality (that's a consequence of the modified budget bound $U + \gamma$). But then $P'(\lambda^* - \delta) > P'(\lambda^*)$, and again λ^* would not maximize (6).

In general, when λ is changing then the corresponding set of job bundles in the corresponding optimal schedule is also changing. By using small perturbations of the job processing times, we may assume that in the point λ^* at most one of the following events happens:

- Two pieces J' and J'' (that may be job bundles or single jobs) are merged into a single job bundle J.
- A job bundle J is split into two pieces J' and J'' (that again may be job bundles or simple jobs).

Now we consider a minimal sequence of transpositions that transforms permutation α into β. In each such transposition some job is swapped with another job. Since α and β both are optimal permutations for $P'(\lambda^*)$, the swapped jobs must have the same Smith ratio $\ell_j^{\lambda^*}/p_j$. We now argue that such a transposition can not change the objective function a lot.

Lemma 4. *Every transposition increases* $\sum_j w_j C_j$ *by at most* $2\,\varepsilon\,\mathtt{OPT}$.

Proof. At least one of the two swapped jobs is not a job bundle (since the ordering of job bundles is fully determined by the precedence constraints `Chain`).

First we consider the case where both swapped jobs i and i' are single jobs (and not job bundles), Without loss of generality we assume that in schedule α

job i is scheduled before job i'. Then, the swap increases $\sum_j w_j p_j$ by an amount of $w_i p_{i'} - p_i w_{i'}$. Since (i, i') is not an active critical pair, this increase is at most $\varepsilon\,\mathtt{OPT}$.

Next, consider the case where a single job i is swapped with a job bundle J. Since the schedules α and β place i and J in different relative orders, we conclude that job i can not be a dangerous job.

- If α runs job i before J, then the swap changes $\sum_j w_j C_j$ by $w_i p_J - p_i w_J$.
- If α runs job i after J, then the swap changes $\sum_j w_j C_j$ by $w_J p_i - p_J w_i$.

In any case, the objective value $\sum_j w_j C_j$ increases by at most $|w_i p_J - p_i w_J|$. We now prove that increase is bounded by $|w_i p_J - p_i w_J| \leq 2\,\varepsilon\,\mathtt{OPT}$. We distinguish three subcases.

In the first subcase, we assume that the job bundle J does not change in λ^*. Since i is not a misplaced dangerous job, we read from (5) that $|w_i p_J - p_i w_J| \leq \tau \leq \varepsilon\,\mathtt{OPT}$. This yields the desired inequality.

In the second subcase, we assume that the job bundle J is created in λ^* by merging two pieces J' and J'' (that were job bundles for some $\lambda = \lambda^* - \delta$ sufficiently close to λ^*). In this case we have

$$\begin{aligned}
|w_i P_J - p_i w_J| &= |w_i P_{J'} - p_i w_{J'} + w_i P_{J''} - p_i w_{J''}| \\
&\leq |w_i P_{J'} - p_i w_{J'}| + |w_i P_{J''} - p_i w_{J''}| \\
&\leq 2\tau \\
&\leq 2\varepsilon\mathtt{OPT}.
\end{aligned}$$

In the third subcase, we assume that the job bundle J results from splitting a piece J' in λ^* (that is: for $\lambda < \lambda^*$, piece J' is a job bundle, and for some $\lambda = \lambda^* + \delta$ sufficiently close to λ^*, piece J' is replaced with two pieces J and J''). In this case we have

$$\begin{aligned}
|w_i p_J - p_i w_J| &= |w_i(p_{J'} - p_{J''}) - p_i(w_{J'} - w_{J''})| \\
&= |(w_i p_{J'} - p_i w_{J'}) - (w_i p_{J''} - p_i w_{J''})| \\
&\leq |w_i p_{J'} - p_i w_{J'}| + |w_i p_{J''} - p_i w_{J''}| \\
&\leq 2\tau \\
&\leq 2\varepsilon\mathtt{OPT}.
\end{aligned}$$

Since we have settled all possible cases and subcases, the proof is complete. □

Along the minimal sequence of transpositions that transforms permutation α into β, we return the first permutation that satisfies the budget constraint $\sum_j u_j C_j \leq U$. Every permutation in this sequence constitutes an optimal solution with respect to the job weights ℓ^{λ^*}. The permutation that comes just before the returned permutation satisfies $\sum_j w_j C_j \leq \mathtt{OPT}$. By Lemma 4, the swapping of jobs leading to the next permutation increases the w-cost by at most $2\,\varepsilon\,\mathtt{OPT}$. Hence, the returned solution has an objective value of at most $(1+2\varepsilon)\,\mathtt{OPT}$. (And if we take into account that because of our first guessing step we are not working

with the exact value of OPT, but only with an approximation thereof, then this bound becomes the slightly weaker bound $(1+2\varepsilon)(1+\varepsilon)\,\mathtt{OPT}$). In any case, we have established the following result.

Theorem 2. *Problem CWSCT has a polynomial time approximation scheme.*

4 Something about Smith and Horn

In a classical paper in scheduling theory, Smith [10] describes a polynomial time algorithm for the problem of minimizing the sum of weighted job completion times on a single machine (without any precedence constraints): A simple exchange argument shows that the jobs should be ordered by non-increasing Smith ratios w_j/p_j.

In another classical paper, Horn [4] gives a polynomial time solution for the problem of minimizing the sum of weighted job completion times on a single machine under *tree-like* precedence constraints. In this paper, we are only interested in the following highly restricted special case: There are two job families $K_1,\ldots,K_k$ and $L_1,\ldots,L_m$. The jobs in the first family are totally ordered by the single chain $K_1 \to K_2 \to \cdots \to K_k$. The jobs in the second family are all independent of each other, and independent of the jobs in the first family.

Suppose that there are two consecutive jobs $K_a \to K_b$ in the chain with $w_a/p_a < w_b/p_b$ (that is, they ordered against their Smith ratios). Then, we may assume that no other job L_c is scheduled between these two jobs: If $w_c/p_c < w_b/p_b$, then L_c is better swapped with K_b, and if $w_c/p_c > w_a/p_a$, then L_c is better swapped with K_a. (A similar swapping argument works, in case there are more than one jobs scheduled between K_a and K_b.) Hence, in any optimal schedule K_a and K_b are consecutive, and we may merge them into a single new job with processing time p_a+p_b and weight w_a+w_b. Optimal schedules for the new instance translate immediately into optimal schedules for the old instance, and vice versa. Note that the optimal objective value of the new instance equals $w_a p_b$ plus the objective value of the old instance; however, this shift in the objective value does not concern the structure of optimal solutions.

Horn's [4] algorithm repeats this merging operation over and over again, as long as there are consecutive (possibly merged) jobs in the chain that are ordered against their Smith ratios. When this procedure terminates, the chain has been cut into a number of so-called *bundles* of merged jobs. Some bundles may consist of only one job. For every bundle J, we denote by p_J the total processing time of all the jobs in J and by w_J their total weight. Along the chain the bundles are ordered by non-increasing Smith ratios w_J/p_J. Thus the precedence constraints become non-restrictive and disappear. We are left with a number of *job bundles* that result from the chain, and with a number of original (non-bundle, non-chain) jobs.

References

1. J.A. Aslam, A. Rasala, C. Stein, and N.E. Young (1999). Improved bicriteria existence theorems for scheduling. In *Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'1999)*, 846–847.
2. M.R. Garey and D.S. Johnson (1979). *Computers and Intractability.* Freeman, San-Francisco.
3. M.X. Goemans and R. Ravi (1996). The constrained minimum spanning tree problem. *Proceedings of the 5th Scandinavian Workshop on Algorithm Theory (SWAT'1996)*, LNCS 1097, Springer Verlag, 66–75.
4. W.A. Horn (1972). Single-machine job sequencing with tree-like precedence ordering and linear delay penalties. *SIAM Journal on Applied Mathematics 23*, 189–202.
5. J.-H. Lin and J.S. Vitter (1992). ε-approximations with minimum packing violation. *Proceedings of the 24th Annual ACM Symposium on the Theory of Computing (STOC'1992)*, 771–782.
6. S.T. McCormick and M.L. Pinedo (1995). Scheduling n independent jobs on m uniform machines with both flowtime and makespan objectives: a parametric analysis. *ORSA Journal on Computing 7*, 63–77.
7. A. Rasala, C. Stein, E. Torng, and P. Uthaisombut (2002). Existence theorems, lower bounds and algorithms for scheduling to meet two objectives. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'2002)*, 723–731.
8. R. Ravi (2002). Bicriteria spanning tree problems. *Proceedings of the 5th Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX'2002)*, LNCS 2462, Springer Verlag, 3–4.
9. D.B. Shmoys and E. Tardos (1993). An approximation algorithm for the generalized assignment problem. *Mathematical Programming 62*, 461–474.
10. W.E. Smith (1956). Various optimizers for single-stage production. *Naval Research Logistics Quarterly 3*, 59–66.
11. C. Stein and J. Wein (1997). On the existence of schedules that are near-optimal for both makespan and total-weighted completion time. *Operations Research Letters 21*, 115–122.