Available online at www.sciencedirect.com
SCIENCE DIRECT®
ELSEVIER
Discrete Applied Mathematics 131 (2003) 237–252
DISCRETE APPLIED MATHEMATICS
www.elsevier.com/locate/dam

On the approximability of average completion time scheduling under precedence constraints☆

Gerhard J. Woeginger[a,b,*,1]

[a] *Institut für Mathematik, Technische Universität Graz, Steyrergasse 30, 8010 Graz, Austria*
[b] *Department of Mathematics, University of Twente, P.O. Box 217, 7500 AE Enschede, Netherlands*

Received 14 November 2000; received in revised form 4 July 2001; accepted 5 May 2002

Abstract

We consider the scheduling problem of minimizing the average weighted job completion time on a single machine under precedence constraints. We show that this problem with arbitrary job weights, the special case of the problem where all job weights are one, and several other special cases of the problem all have the same approximability threshold with respect to polynomial time approximation algorithms. Moreover, for the special case of interval order precedence constraints and for the special case of convex bipartite precedence constraints, we give a polynomial time approximation algorithm with worst case performance guarantee arbitrarily close to the golden ratio $\frac{1}{2}(1+\sqrt{5}) \approx 1.61803$.

Keywords: Scheduling; Precedence constraints; Interval order; Bipartite order; Approximation algorithms; Approximability threshold

1. Introduction

We consider the problem of scheduling n jobs on a single machine. Each job J_j $(j=1,\ldots,n)$ is specified by its length p_j and its weight w_j, where p_j and w_j are non-negative integers. Each job J_j must be scheduled for p_j units of time, and only one job can be scheduled at any point in time. We only consider *non-preemptive* schedules, in which all p_j time units of job J_j must be scheduled consecutively. Precedence

☆ An extended abstract of this paper has appeared in the Proceedings of the 28th International Colloquium on Automata, Languages and Programming.

* Department of Mathematics, University of Twente, P.O. Box 217, 7500 AE Enschede, Netherlands.
E-mail address: g.j.woeginger@math.utwente.nl (G.J. Woeginger).

[1] Supported by the START program Y43-MAT of the Austrian Ministry of Science.

0166-218X/03/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
PII: S0166-218X(02)00427-4

constraints are given by a partial order on the jobs; if J_i precedes J_j in the partial order (denoted by $J_i \rightarrow J_j$), then J_i must be processed before J_j can begin its processing. In this situation, job J_i is called a *predecessor* of J_j, and J_j is called a *successor* of J_i. Our goal is to find a schedule which minimizes the sum $\sum_{j=1}^{n} w_j C_j$ where C_j is the time at which job J_j completes in the given schedule. In the standard three-field scheduling notation (see, e.g. [5,8]) this problem is denoted by $1 \mid prec \mid \sum w_j C_j$, and the special case where $w_j \equiv 1$ is denoted by $1 \mid prec \mid \sum C_j$. Both problems $1 \mid prec \mid \sum w_j C_j$ and $1 \mid prec \mid \sum C_j$ are NP-hard in the strong sense [7,9].

A polynomial time *ρ-approximation algorithm* is a polynomial time algorithm that always returns a near-optimal solution with cost at most a factor ρ above the optimal cost (where $\rho > 1$ is some fixed real number). The value ρ is called the *worst case performance guarantee* of the approximation algorithm. A family of polynomial time $(1+\varepsilon)$-approximation algorithms over all $\varepsilon > 0$ is called a *polynomial time approximation scheme* (PTAS). The *approximability threshold* of a minimization problem is the infimum of all values ρ for which the problem possesses a polynomial time ρ-approximation algorithm. Hence, if a minimization problem has a PTAS then its approximability threshold equals one.

There are several different polynomial time 2-approximation algorithms known for the problem $1 \mid prec \mid \sum w_j C_j$. Hall et al. [6] give such a 2-approximation algorithm by using linear programming relaxations. Chudak and Hochbaum [2] design another 2-approximation algorithm that is based on a half integral linear programming relaxation and on a minimum cut computation in an underlying network. Independently of each other, Margot et al. [10] and Chekuri and Motwani [1] provide (identical) extremely simple, combinatorial polynomial time 2-approximation algorithms. It is an outstanding open problem to determine the exact approximability threshold of $1 \mid prec \mid \sum w_j C_j$; see, e.g. [14]. According to the current state of the art, this threshold might be any value between 1 (which would mean: the problem has a PTAS) and 2 (which would mean: the currently known approximation algorithms are already best possible).

Interestingly, the special case $1 \mid prec \mid \sum C_j$ seems to be no easier to approximate than the general case $1 \mid prec \mid \sum w_j C_j$. The best approximation algorithms known for $1 \mid prec \mid \sum C_j$ have performance guarantee 2, i.e., exactly the same guarantee as for the general case. Every constructive approach that works for the special case seems to carry over to the general case. In this paper, we will show that the cumulative experiences of the research community with these two problems are not just a coincidence:

Theorem 1. *The approximability thresholds of the following eight special cases of the scheduling problem* $1 \mid prec \mid \sum w_j C_j$ *all coincide*:

(a) *The approximability threshold* τ_a *of the general problem* $1 \mid prec \mid \sum w_j C_j$.
(b) *The approximability threshold* τ_b *of the special case where* $1 \leqslant w_j \leqslant n^2$ *and* $1 \leqslant p_j \leqslant n^2$.
(c) *The approximability threshold* τ_c *of the special case where* $w_j \equiv 1$.
(d) *The approximability threshold* τ_d *of the special case where* $p_j \equiv 1$.
(e) *The approximability threshold* τ_e *of the special case where* $w_j \equiv 1$ *and* $p_j \in \{0,1\}$.

(f) *The approximability threshold* τ_f *of the special case where* $w_j \in \{0,1\}$ *and* $p_j \equiv 1$.
(g) *The approximability threshold* τ_g *of the special case where every job has either* $p_j = 0$ *and* $w_j = 1$, *or* $p_j = 1$ *and* $w_j = 0$.
(h) *The approximability threshold* τ_h *of the special case where every job has either* $p_j = 0$ *and* $w_j = 1$, *or* $p_j = 1$ *and* $w_j = 0$, *and where the existence of a precedence constraint* $J_i \to J_j$ *implies that* $p_i = 1$ *and* $w_i = 0$, *and that* $p_j = 0$ *and* $w_j = 1$.

Chekuri and Motwani [1] design instances of the restricted form (h) to show that the integrality gap of a linear ordering relaxation of Potts [13] is 2. Our Theorem 1 gives some insight why even these highly restricted special cases yield worst possible integrality gaps. Hall et al. [6] observe that the time-indexed linear programming formulation of Dyer and Wolsey [3] has an integrality gap of at most 2. One problem with this 2-relaxation is that its size is proportional to the overall job length, which in general is not polynomially bounded in the input size. Hall et al. [6] find a way around this problem by using an equivalent interval-indexed formulation. Theorem 1 offers another way around this problem, since it shows that the thresholds for problems (a) and (b) are equivalent with respect to approximation, and in problem (b) the overall job length is polynomially bounded in the input size.

Our second main result is that for the special case of interval order precedence constraints and for the special case of convex bipartite precedence constraints, one can get an improved performance guarantee that is arbitrarily close to the golden ratio $\frac{1}{2}(1+\sqrt{5})$. For the exact definitions of these two special classes of precedence constraints we refer the reader to the first paragraphs in sections 4.1 and 4.2, respectively.

Theorem 2. *Let* $\phi = \frac{1}{2}(1+\sqrt{5}) \approx 1.61803$ *be the positive real root of* $\phi^2 = \phi + 1$. *For any* $\varepsilon > 0$, *there exists a polynomial time* $(\phi+\varepsilon)$*-approximation algorithm for problem* $1 \mid prec \mid \sum w_j C_j$ *in case* (a) *the precedence constraints form an interval order*, *and in case* (b) *the precedence constraints form a convex bipartite order*.

The paper is organized as follows. In Section 2, we collect several useful definitions and notations, and we also summarize important tools from the literature. In Section 3, we show that the eight approximability thresholds stated in Theorem 1 are all equal. In Section 4, we first show that a polynomial time algorithm for a certain auxiliary problem implies a polynomial time $(\phi+\varepsilon)$-approximation algorithm for $1 \mid prec \mid \sum w_j C_j$. Then we show that for interval orders and for convex bipartite orders this auxiliary problem indeed can be solved in polynomial time. This will prove Theorem 2. Finally, Section 5 contains the conclusions.

2. Definitions, propositions, and preliminaries

For any instance I of $1 \mid prec \mid \sum w_j C_j$, we denote its optimal objective value by $Opt(I)$, and we write $p(I) = \sum_{j=1}^{n} p_j$ and $w(I) = \sum_{j=1}^{n} w_j$. For any subset S of jobs,

let $p(S)=\sum_{J_j\in S} p_j$ and $w(S)=\sum_{J_j\in S} w_j$. A subset S of jobs is called an *initial set* if for every job $J_j\in S$ also all the predecessors of J_j are in S.

Goemans and Williamson [4] provide a nice geometric way of looking at $1\,|\,prec\,|\,\sum w_jC_j$ via a two-dimensional Gantt chart. The Gantt chart is a big rectangle with its four corners at $(0,0)$, $(0,w(I))$, $(p(I),0)$, and $(p(I),w(I))$. The straight line from corner $(0,w(I))$ to corner $(p(I),0)$ is called the *diagonal* of the Gantt chart. Each job J_j is represented by a *job rectangle* of length p_j and height w_j. A schedule $J_{\pi(1)},J_{\pi(2)},\ldots,J_{\pi(n)}$ is represented by the following placement of the job rectangles. The rectangle for the first job $J_{\pi(1)}$ is placed such that its upper left corner lies in $(0,w(I))$. For $j=2,\ldots,n$ the rectangle for $J_{\pi(i)}$ is placed such that its upper left corner coincides with the lower right corner of the rectangle for $J_{\pi(i-1)}$. Clearly, the rectangle for the last job $J_{\pi(n)}$ then has its lower right corner in $(p(I),0)$. It can be seen [4] that the total weighted completion time in the schedule $J_{\pi(1)},J_{\pi(2)},\ldots,J_{\pi(n)}$ equals the total area of the job rectangles plus the total area that lies below these rectangles in the Gantt chart. This total area (that equals the objective value) will be called the *covered area* of the schedule.

Based on the decomposition results of Sidney [15] from 1975 and on the algorithmic results of Lawler [7] from 1978, Margot et al. [10] and Chekuri and Motwani [1] show that within polynomial time, any instance I of $1\,|\,prec\,|\,\sum w_jC_j$ can be split into $k\geqslant 1$ subinstances $I_1,\ldots,I_k$ such that the following two conditions are fulfilled. (i) In any subinstance I_i all initial sets S satisfy $p(S)\,w(I_i)\geqslant p(I_i)w(S)$. (ii) If σ_i $(i=1,\ldots,k)$ is an arbitrary optimal schedule for instance I_i, then putting the schedules $\sigma_1,\sigma_2,\ldots,\sigma_k$ in series yields an optimal schedule for the original instance I. With this, any ρ-approximation algorithm for instances that satisfy condition (i) immediately yields a ρ-approximation algorithm for general instances; simply compute ρ-approximate solutions for the instances $I_1,\ldots,I_k$ and put them in series. Moreover, condition (i) implies that in the Gantt chart of any feasible schedule for instance I_i, the lower right corners of all job rectangles lie on or above the diagonal of the chart; hence, the covered area of such a schedule is at least $w(I_i)\,p(I_i)/2$.

Proposition 3 (Margot et al. [10], Chekuri and Motwani [1]). *With respect to approximation algorithms for* $1\,|\,prec\,|\,\sum w_jC_j$, *we may restrict ourselves to instances* I *that satisfy* $p(S)\,w(I)\geqslant p(I)\,w(S)$ *for any initial set* S. *Such instances* I *a priori satisfy* $Opt(I)\geqslant w(I)\,p(I)/2$.

For any instance I of $1\,|\,prec\,|\,\sum w_jC_j$, we denote by $I^{\#}$ the instance that results from I by replacing every job J_j in I by a new job $J_j^{\#}$ with $w_j^{\#}=p_j$ and $p_j^{\#}=w_j$ and by introducing the precedence constraint $J_j^{\#}\rightarrow J_i^{\#}$ if and only if $J_i\rightarrow J_j$ is in I. In other words, $I^{\#}$ results from I by interchanging job weights and lengths and by reversing the precedence constraints. Instance $I^{\#}$ is called the *reverse instance* of I. Chudak and Hochbaum [2] observe that in case $J_{\pi(1)},J_{\pi(2)},\ldots,J_{\pi(n)}$ is a feasible schedule for instance I, then $J^{\#}_{\pi(n)},\ldots,J^{\#}_{\pi(2)},J^{\#}_{\pi(1)}$ is a feasible schedule for instance $I^{\#}$. It is easily verified that the objective values of these two schedules are the same.

Proposition 4 (Chudak and Hochbaum [2]). *There is a polynomial time computable one-to-one correspondence between feasible solutions of instances I and $I^{\#}$ that preserves objective values.*

An important consequence of Proposition 4 is that any ρ-approximation for instance I trivially translates into a ρ-approximation for the reverse instance $I^{\#}$. We will heavily use this fact throughout the paper. Note that it immediately implies the equalities $\tau_c = \tau_d$ and $\tau_e = \tau_f$ as claimed in Theorem 1.

We now introduce yet another transformation on instances of $1 \mid prec \mid \sum w_j C_j$. Let I be an arbitrary instance of $1 \mid prec \mid \sum w_j C_j$. Then the instance I^+ results from I by splitting every job into several new jobs: For every job J_j with length p_j and weight w_j in instance I, there is a corresponding job K_j^+ in I^+ that has length p_j and weight 0. Moreover, there are w_j other jobs corresponding to J_j that all have length 0 and weight 1; this whole group of w_j jobs is denoted by G_j^+. The precedence constraints in I^+ are defined as follows. $K_i^+ \to K_j^+$ in I^+ if and only if $J_i \to J_j$ in I. All jobs in G_j^+ have job K_j^+ as their common predecessor. In any 'reasonable' schedule for I^+, all jobs in G_j^+ will be processed right after job K_j^+ and thus will form a contiguous block together with job K_j^+ (otherwise, one could decrease the objective value by moving the jobs in G_j^+ directly after job K_j^+). This yields a straightforward one-to-one correspondence between feasible schedules for I and 'reasonable' feasible schedules for I^+: Jobs J_j in a schedule for I may be replaced by their corresponding blocks K_j^+ and G_j^+, and vice versa. Then in I there is one job of weight w_j that completes at a certain time t, whereas in I^+ there are w_j corresponding jobs of weight 1 that all complete at time t.

Proposition 5. *There is a one-to-one correspondence between feasible solutions of instances I and I^+ that preserves objective values.*

3. Equality of the eight approximability thresholds

In this section we will prove Theorem 1 in several steps. The statement of Lemma 6 in Section 3.1 yields $\tau_a \leqslant \tau_b$. Lemmas 7 and 8 in Section 3.2, respectively, yield $\tau_b \leqslant \tau_g$ and $\tau_g \leqslant \tau_h$. Since problem (h) is a special case of problem (a), $\tau_h \leqslant \tau_a$ holds. Summarizing, we have

$$\tau_a \leqslant \tau_b \leqslant \tau_g \leqslant \tau_h \leqslant \tau_a.$$

Therefore, the thresholds for problems (a), (b), (g), and (h) all coincide. Lemma 9 in Section 3.3 yields $\tau_h \leqslant \tau_e$. Since problem (e) is a special case of problem (c), and since problem (c) in turn is a special case of problem (a), we get

$$\tau_a = \tau_h \leqslant \tau_e \leqslant \tau_c \leqslant \tau_a.$$

Therefore, the thresholds for problems (a), (c), (e), and (h) all coincide. Finally, the discussion after Proposition 4 gives $\tau_c = \tau_d$ and $\tau_e = \tau_f$.

To summarize, the four statements in Lemmas 6–9 together indeed will be sufficient to prove all the statements in Theorem 1.

3.1. How case (a) can be reduced to case (b)

Consider an arbitrary instance I of $1 \mid prec \mid \sum w_j C_j$ with $n \geqslant 16$ jobs. To keep the presentation simple, we will write W for $w(I)$ and P for $p(I)$. By Proposition 3 we may assume without loss of generality that

$$Opt(I) \geqslant WP/2. \tag{1}$$

We define another instance I' that results from I by scaling the weights: For every job J_j in I there is a corresponding job J'_j in instance I' with the same length $p'_j = p_j$ and with a new weight of

$$w'_j = \max\{\lceil w_j n^2/W \rceil, 1\} \leqslant n^2. \tag{2}$$

Note that $w'_j \geqslant w_j n^2/W$ and that $w'_j \leqslant w_j n^2/W + 1$. The precedence constraints in I and I' are exactly the same, i.e., $J'_i \to J'_j$ in I' if and only if $J_i \to J_j$ in I. We now assume that for some $\rho \leqslant 2$ we have a ρ-approximate schedule α for instance I' with objective value $A(I') \leqslant \rho\, Opt(I')$ and with job completion times $C'_{\alpha(1)}, \ldots, C'_{\alpha(n)}$. If we use α as an approximate schedule for the unscaled instance I with objective value $A(I)$, then corresponding jobs in the two schedules have the same completion times. This yields

$$A(I') = \sum_{j=1}^{n} w'_{\alpha(j)} C'_{\alpha(j)} \geqslant \frac{n^2}{W} \sum_{j=1}^{n} w_{\alpha(j)} C'_{\alpha(j)} = \frac{n^2}{W} A(I). \tag{3}$$

Next we claim that

$$Opt(I') \geqslant n \log_2 n \cdot P. \tag{4}$$

Suppose for the sake of contradiction that $Opt(I') < n \log_2 n \cdot P$. Then the inequality in (3), the fact that $A(I') \leqslant \rho\, Opt(I')$, and the inequalities $\rho \leqslant 2$ and $n \geqslant 16$ together yield that

$$Opt(I) \leqslant A(I) \leqslant \frac{W}{n^2} A(I') \leqslant \frac{W}{n^2} \rho\, Opt(I') < \frac{W}{n^2} \rho n \log_2 n \cdot P \leqslant WP/2. \tag{5}$$

This blatantly contradicts (1) and thus proves (4). Next, we assume without loss of generality that $J_1, J_2, \ldots, J_n$ is an optimal schedule for instance I with job completion times $C_1, \ldots, C_n$. Then

$$\begin{aligned} Opt(I) = \sum_{j=1}^{n} w_j C_j &\geqslant \sum_{j=1}^{n} W(w'_j - 1) C_j / n^2 = \frac{W}{n^2} \left(\sum_{j=1}^{n} w'_j C_j - \sum_{j=1}^{n} C_j \right) \\ &\geqslant \frac{W}{n^2} (Opt(I') - nP) \geqslant \frac{W}{n^2} Opt(I')(1 - 1/\log_2 n). \end{aligned} \tag{6}$$

Here we first used that $w'_j \leqslant w_j n^2/W + 1$, then that $C_j \leqslant P$, and finally the inequality in (4). By combining (3), (6), and $A(I') \leqslant \rho\, Opt(I')$ we conclude

that

$$A(I)/Opt(I) \leqslant (W \cdot A(I')/n^2)/(W \cdot Opt(I')(1-1/\log_2 n)/n^2)$$
$$= A(I')/(Opt(I')(1-1/\log_2 n)) \leqslant \rho/(1-1/\log_2 n). \quad (7)$$

Lemma 6. *Assume that there is a polynomial time ρ-approximation algorithm A for the special case* (b) *of* $1 \mid prec \mid \sum w_j C_j$ *where* $1 \leqslant w_j \leqslant n^2$ *and* $1 \leqslant p_j \leqslant n^2$ *holds for all jobs* J_j. *Then for every* $\varepsilon > 0$ *there exists a polynomial time* $(\rho+\varepsilon)$*-approximation algorithm for the general problem* $1 \mid prec \mid \sum w_j C_j$.

Proof. Consider an arbitrary instance I of $1 \mid prec \mid \sum w_j C_j$ with n jobs. If $n < 2^{4\rho/\varepsilon}$, then the problem is of constant size and we may solve it in constant time by complete enumeration. Otherwise, $n \geqslant 2^{4\rho/\varepsilon}$ and $\rho/(1-1/\log_2 n)^2 \leqslant \rho+\varepsilon$. Let I' result from I by scaling the job weights as described above; then by (2) $1 \leqslant w_j \leqslant n^2$ holds for all jobs in I'. Let $I'^{\#}$ be the reverse instance of I'; then $1 \leqslant p_j \leqslant n^2$ holds for all jobs in $I'^{\#}$. Let $I'^{\#\prime}$ result from $I'^{\#}$ by scaling the job weights as described above; then $1 \leqslant w_j \leqslant n^2$ and $1 \leqslant p_j \leqslant n^2$ hold for all jobs in $I'^{\#\prime}$. Note that the three instances I', $I'^{\#}$, and $I'^{\#\prime}$ can be determined in polynomial time.

We apply the polynomial time ρ-approximation algorithm A to instance $I'^{\#\prime}$, and interpret the resulting approximate schedule $\alpha'^{\#\prime}$ as a schedule for $I'^{\#}$. By (7) this yields an approximate schedule $\alpha'^{\#}$ for $I'^{\#}$ with objective value at most $\rho/(1-1/\log_2 n)$ above $Opt(I'^{\#})$. By Proposition 4, the approximate schedule $\alpha'^{\#}$ can be translated into an approximate schedule α' of the same approximation quality for instance I'. Finally, we interpret the approximate schedule α' as a schedule for I. By applying once again (7), we get that the resulting approximate objective value for I is at most $\rho/(1-1/\log_2 n)^2 \leqslant \rho+\varepsilon$ above $Opt(I)$. This yields the desired polynomial time $(\rho+\varepsilon)$-approximation algorithm for the general problem $1 \mid prec \mid \sum w_j C_j$. □

3.2. How case (b) can be reduced to cases (g) and (h)

In Lemma 7, we will reduce case (b) with polynomially bounded job lengths and polynomially bounded job weights to case (g). In Lemma 8, we will then reduce case (g) to the most restricted and most primitive case (h) with bipartite precedence constraints.

Lemma 7. *Assume that there is a polynomial time ρ-approximation algorithm A for the special case* (g) *of* $1 \mid prec \mid \sum w_j C_j$ *where every job has either* $p_j = 0$ *and* $w_j = 1$, *or* $p_j = 1$ *and* $w_j = 0$. *Then there also exists a polynomial time ρ-approximation algorithm for the special case* (b) *of* $1 \mid prec \mid \sum w_j C_j$ *where* $1 \leqslant w_j \leqslant n^2$ *and* $1 \leqslant p_j \leqslant n^2$ *holds for all jobs* J_j.

Proof. Consider an arbitrary instance I of $1 \mid prec \mid \sum w_j C_j$ with n jobs where $1 \leqslant w_j \leqslant n^2$ and $1 \leqslant p_j \leqslant n^2$ holds for all jobs. Let I^+ result from I by splitting the jobs as

described in Section 2 just before Proposition 5. Then I^+ contains only jobs of two types: Firstly, jobs with positive length and weight 0, and secondly, jobs with length 0 and weight 1. Since the weights in instance I are polynomially bounded, I^+ can be computed in polynomial time. Next let $I^{+\#}$ be the reverse instance of I^+. Then $I^{+\#}$ contains only jobs of two types: First, jobs with positive weight and length 0, and secondly, jobs with weight 0 and length 1. Finally, let $I^{+\#+}$ result from $I^{+\#}$ by splitting the jobs. Then $I^{+\#+}$ contains only jobs of three types: Jobs with length 0 and weight 0, jobs with length 0 and weight 1, and jobs with length 1 and weight 0. Jobs of length 0 and weight 0 may be disregarded. Note that I^+, $I^{+\#}$, and $I^{+\#+}$ all have the same optimal objective value and all can be determined in polynomial time.

We apply the polynomial time ρ-approximation algorithm A to instance $I^{+\#+}$. We interpret the resulting approximate schedule as a schedule for $I^{+\#}$ by applying Proposition 5, translate it into an approximate schedule for I^+ by applying Proposition 4, and interpret the resulting schedule as an approximate schedule for I by once again applying Proposition 5. As all these translations do not change the objective value, this yields a ρ-approximation for I. □

Lemma 8. *Assume that there is a polynomial time ρ-approximation algorithm A for the special case* (h) *of* $1 \mid prec \mid \sum w_j C_j$ *where every job has either* $p_j = 0$ *and* $w_j = 1$, *or* $p_j = 1$ *and* $w_j = 0$, *and where the existence of a precedence constraint* $J_i \to J_j$ *implies that* $p_i = 1$ *and* $w_i = 0$, *and that* $p_j = 0$ *and* $w_j = 1$. *Then there also exists a polynomial time ρ-approximation algorithm for the special case* (g) *of* $1 \mid prec \mid \sum w_j C_j$ *where every job has either* $p_j = 0$ *and* $w_j = 1$, *or* $p_j = 1$ *and* $w_j = 0$.

Proof. Consider an arbitrary instance I of $1 \mid prec \mid \sum w_j C_j$ of type (g) where every job has either $p_j = 0$ and $w_j = 1$ (such a job will be called a 0-job), or $p_j = 1$ and $w_j = 0$ (such a job will be called a 1-job). We recall that the precedence constraints form a partial order, and hence are transitive. We construct a new instance that results from I by removing all the precedence constraints (i) between 0-jobs, (ii) between 1-jobs, and (iii) from 0-jobs to 1-jobs. The resulting instance I^b has bipartite precedence constraints where the 0-jobs form one class and the 1-jobs form the other class of the bipartition. Therefore, instance I^b is of the type (h).

We will now prove that every feasible schedule for the bipartite instance I^b can be transformed into a feasible schedule for the original instance I without increasing the objective value. Clearly, the statement in the lemma will follow from this.

Hence, consider a feasible schedule for I^b. Renumber the jobs such that this schedule processes the jobs in the ordering $J_1, J_2, \ldots, J_n$ with job completion times $C_1, C_2, \ldots, C_n$. We interpret this schedule as a schedule for the original instance I, and we call a pair of jobs J_i and J_j a *violated* pair, if $J_i \to J_j$ in I and if $i > j$. Consider a violated pair J_i and J_j with the difference $i - j$ as small as possible. Consider an intermediate job J_k with $j < k < i$. If J_k was a predecessor of J_i in I, then $J_k \to J_i \to J_j$, and if J_k was a predecessor of J_j in I, then $J_k \to J_j$. In either case, the jobs J_k and J_j would form another violated pair with $k - j < i - j$. If J_k was a successor of J_j in I, then $J_i \to J_j \to J_k$, and if J_k was a successor of J_i in I, then $J_i \to J_k$. In either case, the jobs J_i and J_k would form another violated pair with $i - k < i - j$. Hence,

all intermediate jobs between J_j and J_i are neither predecessors nor successors of J_i and J_j. We distinguish two cases.

(i) J_i is a 0-job. We remove J_i from the schedule and reinsert it immediately before J_j. By moving this 0-job to an earlier point in time, we will not increase the objective value of the schedule.

(ii) J_i is a 1-job. Since the considered schedule is feasible for I^b, in this case also J_j must be a 1-job. We remove J_j from the schedule and reinsert it immediately after J_i. This increases the completion time of J_j from C_j to C_i, and it decreases the completion time of the $i-j$ jobs $J_{j+1},\ldots,J_i$ all by 1. Note that $i-j \geqslant C_i - C_j$ since there must be $C_i - C_j$ 1-jobs among $J_{j+1},\ldots,J_i$. Hence, by moving J_j we will not increase the objective value of the schedule.

To summarize, in both cases we resolve the violated pair J_i and J_j, we do not create any new violated pairs, and we end up with another feasible schedule for I^b. By repeating this procedure over and over again, we will eventually get rid of all violated pairs without ever increasing the objective value. The resulting schedule will be feasible for the original instance I, exactly as we desired. □

3.3. How case (h) can be reduced to case (e)

In this section, we will mainly recycle ideas and constructions from Sections 3.1 and 3.2. Consider an instance I of type (h) with n jobs where every job has either $p_j = 1$ and $w_j = 0$, or $p_j = 0$ and $w_j = 1$, and where the precedence constraints only go from jobs of the first type to jobs of the second type. Note that $w(I) + p(I) = n$. From instance I we construct another instance I' by modifying the weights exactly as described at the beginning of Section 3.1. The processing times and the precedence constraints remain unchanged, and the weights are computed according to Eq. (2). We define

$$Q = \lceil n^2/W \rceil \leqslant n^2.$$

The resulting instance I' only has two types of jobs: Jobs with $p'_j = 1$ and $w'_j = 1$, and jobs with $p'_j = 0$ and $w'_j = Q$. All precedence constraints only go from jobs of the first type to jobs of the second type. By arguments analogous to those used in Lemma 6, any polynomial time ρ-approximation algorithm for instances I' translates into a polynomial time $(\rho+\varepsilon)$-approximation algorithm for instances I.

Next, we split every job J'_j with $p'_j = 0$ and $w'_j = Q$ in I' into Q new jobs that all have length 0 and weight 1. These Q new jobs have exactly the same predecessors as job J'_j in instance I'. The jobs with $p'_j = 1$ and $w'_j = 1$ remain unchanged. The resulting instance I'' can be found in polynomial time. Instance I'' only has two types of jobs: Jobs with $p''_j = 1$ and $w''_j = 1$, and jobs with $p''_j = 0$ and $w''_j = 1$. Therefore, instance I'' is of the type (e). By arguments very similar to those used at the end of Section 2 for instances I^+, one can show that there is a one-to-one correspondence between feasible solutions of instances I' and I'' that preserves objective values.

Lemma 9. *Assume that there is a polynomial time ρ-approximation algorithm A for the special case* (e) *of* $1 \mid prec \mid \sum w_j C_j$ *where* $w_j \equiv 1$ *and* $p_j \in \{0,1\}$. *Then for every*

$\varepsilon > 0$ there exists a polynomial time $(\rho + \varepsilon)$-approximation algorithm for the special case (h) *of* $1 \mid prec \mid \sum w_j C_j$ *where every job has either* $p_j = 0$ *and* $w_j = 1$, *or* $p_j = 1$ *and* $w_j = 0$, *and where the existence of a precedence constraint* $J_i \to J_j$ *implies that* $p_i = 1$ *and* $w_i = 0$, *and that* $p_j = 0$ *and* $w_j = 1$.

4. Approximation algorithms for nice precedence constraints

In this section, we will derive a polynomial time $(\phi + \varepsilon)$-approximation for $1 \mid prec \mid \sum w_j C_j$ for certain 'nice' classes of precedence constraints that include interval orders and convex bipartite orders. This approximation algorithm is based on exact algorithms for the following auxiliary problem.

Problem. Good Initial Set

Instance. An instance I of $1 \mid prec \mid \sum w_j C_j$, i.e., a set of precedence constrained jobs J_j $(j = 1, \ldots, n)$ with non-negative integer lengths p_j and non-negative integer weights w_j. A real number γ with $0 < \gamma \leqslant 1/2$.

Question. Does there exist an initial set T that simultaneously satisfies $p(T) \leqslant (1/2 + \gamma)\, p(I)$ and $(1/2 - \gamma) w(I) \leqslant w(T)$?

Theorem 10. *Let $\mathscr{C}$ be a class of precedence constraints such that the restriction of the Good Initial Set problem to precedence constraints from class $\mathscr{C}$ is solvable in polynomial time. Then for any $\varepsilon > 0$, the restriction of problem $1 \mid prec \mid \sum w_j C_j$ to precedence constraints from class $\mathscr{C}$ has a polynomial time $(\phi + \varepsilon)$-approximation algorithm.*

Proof. Consider an instance I of $1 \mid prec \mid \sum w_j C_j$ with precedence constraints from class $\mathscr{C}$. We will write W short for $w(I)$, and P short for $p(I)$. By Proposition 3 we may assume that $p(S)\, W \geqslant w(S) P$ holds for any initial set S in I, and that $Opt(I) \geqslant WP/2$. Take an arbitrary real ε with $0 < \varepsilon < 1/4$ and define $\ell = \lceil 2\phi/\varepsilon \rceil$. For this choice of ℓ, the following inequalities are satisfied for all γ with $0 < \gamma \leqslant 1/2$:

$$\frac{1}{2} + 2\left(\gamma - \frac{1}{2\ell}\right)^2 \geqslant \gamma \geqslant \frac{2\phi}{\varepsilon\ell}\,\gamma - \frac{\phi}{2\varepsilon\ell^2} = 2\frac{\phi}{\varepsilon}\left(\frac{\gamma}{\ell} - \frac{1}{4\ell^2}\right).$$

Here the first inequality follows from $\gamma \leqslant 1/2$ and the second inequality follows from $2\phi \leqslant \varepsilon\ell$. Rewriting this inequality yields

$$\frac{1}{2} + 2\left(\gamma - \frac{1}{2\ell}\right)^2 \geqslant \left(\frac{1}{2} + 2\gamma^2\right)/(1 + \varepsilon/\phi). \tag{8}$$

We call the polynomial time algorithm for problem Good Initial Set on the inputs I and $\gamma_k = k/2\ell$ for $k = 1, 2, \ldots, \ell$ until we detect a value $\gamma = \gamma_k$ that yields a YES-instance. Let T be the corresponding initial set of jobs, and let U be the set of remaining jobs that are not contained in T. We construct an approximate solution for the scheduling instance I that first runs all the jobs in T in any feasible order, and then runs all the

jobs in U in any feasible order. We denote the objective value of this approximate solution by $A(I)$. Clearly, this approximate solution can be determined in polynomial time.

Now consider the two-dimensional Gantt chart for the approximate schedule (see the discussion in Section 2). Since $p(T) \leqslant (1/2+\gamma)P$, all rectangles for jobs in T lie to the left of the line $x=(1/2+\gamma)P$. Moreover, $w(T) \geqslant (1/2-\gamma)W$ implies $w(U) \leqslant (1/2+\gamma)W$, and thus all rectangles for jobs in U lie below the line $y=(1/2+\gamma)W$. To summarize, not a single job rectangle protrudes into the 'forbidden' region that lies to the right of the line $x=(1/2+\gamma)P$ and above the line $y=(1/2+\gamma)W$. Since the area of this forbidden region equals $(1/2-\gamma)^2\,WP$, the remaining area in the chart that may contribute to the objective value of the approximate schedule is at most $WP-(1/2-\gamma)^2\,WP$. This yields

$$A(I) \leqslant (3/4+\gamma-\gamma^2)WP. \tag{9}$$

Now consider the two-dimensional Gantt chart for an optimal schedule for instance I. Since we assumed by Proposition 3 that $p(S)W \geqslant w(S)P$ holds for any initial set S, in this Gantt chart the whole region below the diagonal must belong to the covered area. Moreover, we claim that the covered area in the Gantt chart must include the rectangular region $\mathscr{R}$ that lies to the left of the vertical line $x=(1/2+\gamma-1/2\ell)P$, and below the horizontal line $y=(1/2+\gamma-1/2\ell)W$. If $\gamma=1/2\ell$ holds then $\mathscr{R}$ lies below the diagonal of the Gantt chart, and our claim trivially holds. Therefore, we may assume $\gamma \geqslant 1/\ell$, in which case the instance of Good Initial Set for the value $\gamma-1/2\ell$ was a NO-instance. If the region $\mathscr{R}$ did not belong to the covered area, then for some job J_z the lower right corner of its rectangle would lie inside of $\mathscr{R}$. But then the initial set that consists of the first job up to job J_z in the optimal schedule would constitute a good initial set for the value $\gamma-1/2\ell$. This contradiction proves the claim. To summarize, the covered area in the Gantt chart for the optimal schedule includes the region below the diagonal together with the region $\mathscr{R}$. This yields

$$Opt(I) \geqslant WP/2+2(\gamma-1/2\ell)^2WP \geqslant (1/2+2\gamma^2)WP/(1+\varepsilon/\phi). \tag{10}$$

Here the first inequality follows from the area estimation, and the second inequality follows from (8). By combining (9) and (10), we conclude that

$$\frac{A(I)}{Opt(I)} \leqslant (1+\varepsilon/\phi)\frac{3/4+\gamma-\gamma^2}{1/2+2\gamma^2} \leqslant (1+\varepsilon/\phi)\phi = \phi+\varepsilon. \tag{11}$$

The final inequality follows by elementary calculus: On the interval $(0,1/2]$, the function $f(\gamma)=(3/4+\gamma-\gamma^2)/(1/2+2\gamma^2)$ takes its maximum value at $\gamma=\frac{1}{2}(\sqrt{5}-2)$, and this maximum value equals $\phi=\frac{1}{2}(1+\sqrt{5})$. The proof is complete. □

4.1. Interval order precedence constraints

An *interval order* (see, e.g. [12]) on the jobs $J_1,\ldots,J_n$ is specified by a set of n intervals $\mathscr{I}_1,\ldots,\mathscr{I}_n$ along the real line. Then $J_i \to J_j$ holds if and only if interval $\mathscr{I}_i$ lies completely to the left of interval $\mathscr{I}_j$. Without loss of generality we assume that all intervals have non-zero length, and that the n intervals have $2n$ pairwise distinct left and right endpoints. Our goal in this subsection is to prove Theorem 2(a).

Lemma 11. *Consider the subset of instances I of $1 \mid prec \mid \sum w_j C_j$ with n jobs for which $w_j \leqslant n^2$ and $p_j \leqslant n^2$ holds for all jobs and for which the precedence constraints form an interval order. The restriction of the Good Initial Set problem to instances (I,γ) where I is from this subset is solvable in polynomial time.*

Proof. Consider such an instance I and a real number γ with $0 < \gamma \leqslant 1/2$. Now any initial set T for an interval order can be decomposed into three parts: The first part is the interval in T with the rightmost left endpoint; this interval is called the *head* $h(T)$ of T. The second part consists of all predecessors of $h(T)$ in I; note that this part is fully specified as soon as the head $h(T)$ is specified. The third part consists of the remaining intervals in T. All these remaining intervals contain the left endpoint of $h(T)$ in their interior, and thus are pairwise incomparable as well as incomparable to $h(T)$.

We divide the problem of finding a good initial set T into n subproblems depending on the head $h(T)$ of T. For every fixed head $h(T)$, the good initial set problem can be solved in polynomial time as follows. We define P^* to be $(1/2+\gamma)p(I)$ minus the length of $h(T)$ minus the overall length of all predecessors of $h(T)$. We define W^* to be $(1/2-\gamma)w(I)$ minus the weight of $h(T)$ minus the overall weight of all predecessors of $h(T)$. Then the existence of a good initial set with head $h(T)$ boils down to deciding whether there exists a subset U of (pairwise incomparable) intervals that all contain the left endpoint of $h(T)$ in their interior, such that $p(U) \leqslant P^*$ and $w(U) \geqslant W^*$. But that is just a standard two-dimensional knapsack problem where the weights and lengths of the elements are polynomially bounded! This special case of the knapsack problem can be solved in polynomial time by the standard dynamic programming approaches (cf., e.g. [11]). □

Proof of Theorem 2(a). By Theorem 10 and Lemma 11, there exists a polynomial time $(\phi+\varepsilon)$-approximation algorithm for the special case of $1 \mid prec \mid \sum w_j C_j$ where $w_j \leqslant n^2$ and $p_j \leqslant n^2$ and where the precedence constraints form an interval order. By Lemma 6, this approximability result carries over to the special case of $1 \mid prec \mid \sum w_j C_j$ under interval order precedence constraints with *arbitrary* job lengths and job weights. Lemma 6 can be applied, since its proof only reverses the precedence constraints, and since the reverse of an interval order is again an interval order. □

We stress that we cannot apply Lemmas 9 and 7 in the above proof of Theorem 2(a). When we split jobs in an interval order as we need to do in the proofs of these two Lemmas, then the resulting order will not necessarily be again an interval order. We also note that the statement in Lemma 11 does not generalize to arbitrary job weights and job lengths unless $P = NP$: In this general case the Good Initial Set problem for interval orders is NP-complete (this can be proved by a reduction from the NP-complete Partition problem). For series-parallel precedence constraints, the situation looks much better: Reversing series-parallel precedence constraints yields again series-parallel precedence constraints, and the splitting of jobs translates series-parallel precedence constraints into new series-parallel precedence constraints. Hence, Theorem 1 yields that approximating $1 \mid prec \mid \sum w_j C_j$ under series-parallel precedence

constraints is equivalent to approximating $1 \mid prec \mid \sum w_j C_j$ under series-parallel precedence constraints when, e.g. $w_j \equiv 1$ and $p_j \in \{0,1\}$. For this special case the Good Initial Set problem is polynomially solvable, and thus Theorem 10 yields the existence of a polynomial time $(\phi+\varepsilon)$-approximation algorithm for $1 \mid prec \mid \sum w_j C_j$ under series-parallel precedence constraints. However, this statement is neither surprising nor interesting, since Lawler [7] has shown that $1 \mid prec \mid \sum w_j C_j$ under series-parallel can even be solved in polynomial time.

4.2. Convex bipartite precedence constraints

Our goal in this subsection is to prove Theorem 2(b). A *bipartite order* (see [12]) on a set of jobs is defined as follows. The jobs are classified into two types. The *minus-jobs* $J_1^-,\ldots,J_a^-$ do not have any predecessors, and the *plus-jobs* $J_1^+,\ldots,J_b^+$ do not have any successors. The only precedence constraints are of the type $J_i^- \rightarrow J_j^+$, that is from minus-jobs to plus-jobs. In this section we are interested in the class of convex bipartite orders. This class forms a proper subset of the class of general bipartite orders, and it is a proper superset of the class of strong bipartite orders (see [12,16] for more information). A bipartite order is a *convex bipartite order* if for every $j=1,\ldots,b$ there exist two indices $\ell(j)$ and $r(j)$ such that $J_i^- \rightarrow J_j^+$ holds if and only if $\ell(j) \leqslant i \leqslant r(j)$. In other words, the predecessor set of every plus-job forms an interval within the minus-jobs.

Lemma 12. *Consider the subset of instances I of $1 \mid prec \mid \sum w_j C_j$ with n jobs for which $w_j \leqslant n^2$ and $p_j \leqslant n^2$ holds for all jobs and for which the precedence constraints form a convex bipartite order. The restriction of the Good Initial Set problem to instances (I,γ) where I is from this subset is solvable in polynomial time.*

Proof. Consider such an instance I. Without loss of generality we assume that every plus-job J_j^+ has at least one predecessor and thus satisfies $\ell(j) \leqslant r(j)$. Moreover, we will assume that the plus-jobs are ordered in such a way that $r(j) \leqslant r(j+1)$ holds for $1 \leqslant j \leqslant b-1$. Finally, we assume that the first plus-job J_1^+ has J_1^- as its unique predecessor and that this first minus-job J_1^- has J_1^+ as its unique successor. The length and the weight of these two jobs are all zero. The jobs J_1^- and J_1^+ serve as dummy jobs that help us to avoid special treatment of certain boundary cases. We will write p_i^- and w_i^- for the length and the weight of job J_i^-, and p_j^+ and w_j^+ for the length and the weight of job J_j^+.

We will now design a polynomial time algorithm for the Good Initial Set problem. This algorithm is based on a Boolean predicate $A[h,i,j,p,w]$ where $1 \leqslant h \leqslant i \leqslant a$, $1 \leqslant j \leqslant b$ such that $r(j) \leqslant i$, and $0 \leqslant p \leqslant p(I)$ and $0 \leqslant w \leqslant w(I)$. Since we assumed $w_j \leqslant n^2$ and $p_j \leqslant n^2$, we have $w(I) \leqslant n^3$ and $p(I) \leqslant n^3$, and thus the total number of these values $A[h,i,j,p,w]$ is polynomially bounded in n. The predicate $A[h,i,j,p,w]$ is true if and only if there exists an initial set T that fulfills the following four conditions.

- T only contains jobs from $J_1^-,\ldots,J_i^-$ and from $J_1^+,\ldots,J_j^+$.
- T contains all $i-h+1$ jobs $J_h^-,\ldots,J_i^-$.

- If $h \geqslant 2$, then T does not contain the job J_{h-1}^-.
- $p(T) = p$ and $w(T) = w$.

Such a set T will be called a *witness* for $A[h,i,j,p,w]$. Clearly, every non-empty initial set T is a witness to some $A[h,i,j,p(T),w(T)]$. Now consider an initial set T that is a witness for $A[h,i,j,p,w]$ with $i \geqslant 2$ and $j \geqslant 2$.

- If T does not contain the job J_j^+, then T is also a witness for $A[h,i,j-1,p,w]$. Any witness for $A[h,i,j-1,p,w]$ is also a witness for $A[h,i,j,p,w]$.
- If T does contain the job J_j^+, if $r(j) < i$, and if $h < i$, then $T - \{J_i^-\}$ is a witness for $A[h,i-1,j,p-p_i^-,w-w_i^-]$. Any witness for $A[h,i-1,j,p-p_i^-,w-w_i^-]$ can be extended to a witness for $A[h,i,j,p,w]$ by adding the job J_i^- to it.
- If T does contain the job J_j^+, if $r(j) < i$, and if $h = i$, then $T - \{J_i^-\}$ is a witness for $A[h',i',j,p-p_i^-,w-w_i^-]$ for some indices h' and i' with $1 \leqslant h' \leqslant i' \leqslant i-2$. Any witness for $A[h',i',j,p-p_i^-,w-w_i^-]$ with $1 \leqslant h' \leqslant i' \leqslant i-2$ can be extended to a witness for $A[h,i,j,p,w]$ by adding the job J_i^- to it.
- If T does contain the job J_j^+, and if $r(j) = i$, then the inequality $h \leqslant \ell(j)$ must be satisfied. In this case, $T - \{J_j^+\}$ is a witness for $A[h,i,j-1,p-p_j^+,w-w_j^+]$. Moreover, any witness for $A[h,i,j-1,p-p_j^+,w-w_j^+]$ can be extended to a witness for $A[h,i,j,p,w]$ by adding the job J_j^+ to it.

Clearly, these four cases cover all possibilities for a witness T for $A[h,i,j,p,w]$ with $2 \leqslant j$ and $2 \leqslant i$. The cases $A[h,i,j,p,w]$ for $i = 1$, and for $j = 1$ and $i = 2$ are trivial, since the dummy jobs J_1^- and J_1^+ restrict their structure; e.g., the only defined case with $i = 1$ is $A[1,1,1,0,0]$. It remains to discuss witnesses T for the case where $j = 1$ and $i \geqslant 3$. Since J_1^+ is a dummy job, we may assume without loss of generality that T only contains minus-jobs in this case.

- If $h < i$, then $T - \{J_i^-\}$ is a witness for $A[h,i-1,1,p-p_i^-,w-w_i^-]$. Any witness for $A[h,i-1,1,p-p_i^-,w-w_i^-]$ can be extended to a witness for $A[h,i,1,p,w]$ by adding the job J_i^- to it.
- If $h = i$, then $T - \{J_i^-\}$ is a witness for $A[h',i',1,p-p_i^-,w-w_i^-]$ for some h' and i' with $1 \leqslant h' \leqslant i' \leqslant i-2$. Any witness for $A[h',i',1,p-p_i^-,w-w_i^-]$ with $1 \leqslant h' \leqslant i' \leqslant i-2$ can be extended to a witness for $A[h,i,1,p,w]$ by adding the job J_i^- to it.

It is straightforward to use the above cases to compute all values $A[h,i,j,p,w]$ by going through them in increasing order of i and j, and by storing all computed values in a table. The overall time needed to compute all values is polynomially bounded in the size of the table. Finally, in order to solve the Good Initial Set problem we only need to scan through the table and look for a true entry $A[h,i,j,p,w]$ with $p \leqslant (1/2+\gamma)p(I)$ and $(1/2-\gamma)w(I) \leqslant w$. □

Proof of Theorem 2(b). The argument is almost identical to the proof of Theorem 2(a) in the preceding subsection. Theorem 10 and Lemma 12 yield the existence of a polynomial time $(\phi+\varepsilon)$-approximation algorithm for the special case where $w_j \leqslant n^2$ and $p_j \leqslant n^2$, and Lemma 6 can be used to carry this over to $1 \mid prec \mid \sum w_j C_j$ under convex bipartite precedence constraints with arbitrary job lengths and arbitrary job weights. (A small technical problem arises, since the proof of Lemma 6 reverses the

precedence constraints, and the reverse of a convex bipartite order is not necessarily again a convex bipartite order. This can be repaired by Proposition 4.)

5. Conclusions

In this paper, we have done a further step in the exploration of the approximability behavior of the scheduling problem $1 \mid prec \mid \sum w_j C_j$. We have shown that the most general version of this problem has the same approximability threshold (with respect to polynomial time approximation algorithms) as some fairly innocent looking special cases of the problem. Moreover, we have shown that for the special cases of interval order precedence constraints and of convex bipartite precedence constraints, the approximability threshold is at most the golden ratio $\phi \approx 1.61803$.

However, it remains an outstanding open problem to find a polynomial time approximation algorithm with worst case performance guarantee strictly better than 2 for the general problem $1 \mid prec \mid \sum w_j C_j$.

Acknowledgements

I thank Yossi Azar, Norbert Blum, Amos Fiat, Anna Karlin, Stefano Leonardi, Petra Schuurman, and Martin Skutella for helpful discussions. I thank an unknown referee for carefully reading a preliminary version of this paper, and for pointing out two major and many minor errors.

References

[1] C. Chekuri, R. Motwani, Precedence constrained scheduling to minimize sum of weighted completion times on a single machine, Discrete Appl. Math. 98 (1999) 29–38.
[2] F. Chudak, D.S. Hochbaum, A half-integral linear programming relaxation for scheduling precedence-constrained jobs on a single machine, Oper. Res. Lett. 25 (1999) 199–204.
[3] M.E. Dyer, L.A. Wolsey, Formulating the single machine sequencing problem with release dates as a mixed integer program, Discrete Appl. Math. 26 (1990) 255–270.
[4] M.X. Goemans, D.P. Williamson, Two-dimensional Gantt charts and a scheduling algorithm of Lawler, SIAM J. Discrete Math. 13 (2000) 281–294.
[5] R.L. Graham, E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, Optimization and approximation in deterministic sequencing and scheduling: a survey, Ann. Discrete Math. 5 (1979) 287–326.
[6] L.A. Hall, A.S. Schulz, D.B. Shmoys, J. Wein, Scheduling to minimize average completion time: off-line and on-line approximation algorithms, Math. Oper. Res. 22 (1997) 513–544.
[7] E.L. Lawler, Sequencing jobs to minimize total weighted completion time subject to precedence constraints, Ann. Discrete Math. 2 (1978) 75–90.
[8] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, D.B. Shmoys, Sequencing and scheduling: algorithms and complexity, in: S.C. Graves, A.H.G. Rinnooy Kan, P.H. Zipkin (Eds.), Logistics of Production and Inventory, Handbooks in Operations Research and Management Science, Vol. 4, North-Holland, Amsterdam, 1993, pp. 445–522.
[9] J.K. Lenstra, A.H.G. Rinnooy Kan, Complexity of scheduling under precedence constraints, Oper. Res. 26 (1978) 22–35.

[10] F. Margot, M. Queyranne, Y. Wang, Decompositions, network flows, and a precedence constrained single machine scheduling problem, Report #2000-29, Department of Mathematics, University of Kentucky, Lexington, 1997.
[11] S. Martello, P. Toth, Knapsack Problems: Algorithms and Computer Implementations, Wiley, England, 1990.
[12] R.H. Möhring, Computationally tractable classes of ordered sets, in: I. Rival (Ed.), Algorithms and Order, Kluwer Academic Publishers, Dordrecht, 1989, pp. 105–193.
[13] C.N. Potts, An algorithm for the single machine sequencing problem with precedence constraints, Math. Programming Stud. 13 (1980) 78–87.
[14] P. Schuurman, G.J. Woeginger, Polynomial time approximation algorithms for machine scheduling: ten open problems, J. Scheduling 2 (1999) 203–213.
[15] J.B. Sidney, Decomposition algorithms for single-machine sequencing with precedence relations and deferral costs, Oper. Res. 23 (1975) 283–298.
[16] J. Spinrad, A. Brandstädt, L. Stewart, Bipartite permutation graphs, Discrete Appl. Math. 18 (1987) 279–292.