Operations Research Letters 25 (1999) 199–204

operations research letters

www.elsevier.com/locate/orms

A half-integral linear programming relaxation for scheduling precedence-constrained jobs on a single machine

Fabián A. Chudak[a,*,1], Dorit S. Hochbaum[b,2]

[a] IBM T.J. Watson Research Center, P.O. Box 218, Yorktown Heights, NY 10598, USA
[b] Department of IEOR, University of California, Berkeley, CA 94720, USA

Received 1 August 1997; received in revised form 1 July 1999

Abstract

We present a new linear programming relaxation for the problem of minimizing the sum of weighted completion times of precedence-constrained jobs. Given a set of n jobs, each job j has processing time p_j and weight w_j. There is also a partial order $\prec$ on the execution of the jobs: if $j \prec k$, job k may not start processing before job j has been completed. For C_j representing the completion time of job j, the objective is to minimize the weighted sum of completion times, $\sum_j w_j C_j$. The new relaxation is simple and compact, has exactly two variables per inequality and half-integral extreme points. An optimal solution can be found via a minimum cut computation, which provides a new 2-approximation algorithm in the complexity of a minimum cut on a graph. As a by-product, we also introduce another new 2-approximation algorithm for the problem.

Keywords: Scheduling; Precedence constraints; Approximation algorithms; Linear programming

1. Introduction

We consider the following nonpreemptive scheduling problem. There are n jobs, $j = 1, \dots, n$, and one machine. Job j is to be processed without interruption for p_j units of time and has a positive weight w_j, $j = 1, \dots, n$. A partial order $\prec$ is imposed on the sequencing of the jobs: $j \prec k$ implies that job k may only start processing once job j has been completed. For a given schedule, let C_j be the completion time of job j, $j = 1, \dots, n$. The objective is to minimize the sum of the weighted completion times, $\sum_{j=1}^{n} w_j C_j$. Using the notation of the survey article of Graham et al. [5], this problem is denoted $1|\text{prec}|\sum w_j C_j$. The problem is known to be NP-complete [8].

The first constant factor guarantee was given by Hall et al. [6], who provided a 2-approximation algorithm. This is the best-known approximation factor to date. The algorithm of Hall et al. is based on solving a linear programming relaxation of the problem. The relaxation uses completion time variables. The linear

* Corresponding author.
E-mail addresses: chudak@watson.ibm.com (F.A. Chudak), dorit@hochbaum.berkeley.edu (D.S. Hochbaum)

[1] This research was conducted while the author was a graduate student at the School of Operations Research and Industrial Engineering, Cornell University, Ithaca, NY 14853, USA. Research partially supported by NSF grants CCR-9307391 and DMI-9157199.

[2] Research supported in part by NSF award No. DMI-9713482 and by SUN Microsystems.

0167-6377/99/$ - see front matter
PII: S0167-6377(99)00056-5

program contains an exponential number of inequalities, yet it can be solved in polynomial time using a separation oracle. Hall et al. also pointed out that a relaxation proposed by Potts [10], a linear ordering formulation, contains only a polynomial number of constraints, and has the property that any feasible solution satisfies all the constraints of their original linear programming relaxation.

In this paper we present a linear programming relaxation that is weaker than that of Potts', but stronger than the one of Hall et al. [6], thus within a factor of 2 from optimum. Our relaxation is based on linear ordering variables, like the linear ordering relaxation of Potts'. However, it contains only 2 variables per inequality and is simpler than Potts' relaxation. In addition, a half-integral optimal solution can be found via a minimum cut computation. This is a consequence of the work of Hochbaum et al. [7].

Finally, we propose a new 2-approximation algorithm for the problem, which is based on a simple observation concerning the interchange of weights and processing times for the general problem.

Independently of our work, a different approach that also provides combinatorial 2-approximation algorithms for the problem was introduced by Chekuri and Motwani [2] and by Margot et al. (the latter was communicated to us by Schulz [13], see also [9]).

2. Linear programming relaxation for $1|\text{prec}|\sum w_j C_j$

2.1. The completion time formulation of Hall et al.

The formulation of Hall et al. [6] uses the variables C_j that represent the completion times of the jobs. Let $N=\{1,\dots,n\}$ be the set of jobs. For each subset of jobs $S\subseteq N$ let $p(S) := \sum_{j\in S} p_j$ and $p^2(S) := \sum_{j\in S} p_j^2$. The following valid inequalities were proposed by Queyranne [11]:

$$\sum_{j\in S} p_j C_j \geqslant \frac{1}{2}(p^2(S) + p(S)^2) \quad \text{for each } S\subseteq N. \quad (1)$$

Note that the right-hand side in (1) is a supermodular function on the subsets of N. More importantly, the right-hand side $\frac{1}{2}(p^2(S) + p(S)^2)$ is equal to the optimal weighted sum of completion times for the set S when each job's weight is equal to its processing time, $\min \sum_{j\in S} p_j C_j$.

The precedence constraints are introduced by adding

$$C_k \geqslant C_j + p_k \quad \text{if } j \prec k. \quad (2)$$

The linear programming relaxation on completion time variables CT can now be written as

$$(\mathsf{CT}) \quad \begin{aligned} &\text{Min} \quad \sum_{j=1}^{n} w_j C_j \\ &\text{s.t.} \quad (1)\text{–}(2) \end{aligned}$$

The 2-approximation algorithm of Hall et al. [6] works as follows: first solve the linear program CT and then schedule the jobs in a sequence corresponding to optimal LP values for $C_j, \bar{C}_j,\ j=1,\dots,n$, in nondecreasing order. Specifically, renaming the jobs so that the optimal solution to the linear program satisfies $\bar{C}_1 \leqslant \cdots \leqslant \bar{C}_n$, inequalities (1) applied to the sets $S=\{1,\dots,j\}$, imply that for each $j=1,\dots,n$,

$$\bar{C}_j \geqslant \frac{1}{2}\sum_{i=1}^{j} p_i.$$

Since the completion time of job j in the schedule produced by the algorithm is $\sum_{i=1}^{j} p_i$, scheduling the jobs in the order $1,\dots,n$ is a 2-approximate solution.

2.2. The linear ordering relaxation of Potts

In the relaxation proposed by Potts [10] there is a binary variable for each pair of jobs i and j, δ_{ij}, which is 1 if i is scheduled before j, and 0 otherwise. Clearly either i is scheduled first or j is, and hence

$$\delta_{ij} + \delta_{ji} = 1, \quad i=1,\dots,n,\ j=1,\dots,n,\ i\neq j. \quad (3)$$

If job i is constrained to precede j in the partial order, then

$$\delta_{ij} = 1 \quad \text{if } i \prec j. \quad (4)$$

To capture the transitivity of a feasible schedule, that is, if i is scheduled before j and j is scheduled before k ($\delta_{kj}=0$), then i must be scheduled before k, the following valid inequalities are used:

$$\delta_{ij} \leqslant \delta_{ik} + \delta_{kj}, \quad i=1,\dots,n,\ j=1,\dots,n,\ k=1,\dots,n,\ i\neq j\neq k\neq i. \quad (5)$$

The completion time of job j, C_j, is

$$C_j = p_j + \sum_{k \neq j} \delta_{kj} p_k, \quad j = 1, \dots, n. \tag{6}$$

It is easy to see that the linear ordering formulation ILO

$$(\mathsf{ILO}) \quad \begin{aligned} \text{Min} \quad & \sum_{j=1}^{n} w_j C_j, \\ \text{s.t.} \quad & (3)\text{–}(6), \\ & \delta_{ij} \in \{0, 1\} \end{aligned}$$

is indeed a complete formulation of the problem (see [16]). To obtain a lower bound, we relax the integrality constraints replacing them by

$$\delta_{ij} \geqslant 0, \quad 1 \neq j, \tag{7}$$

we will refer to the relaxation of ILO as LO.

This linear programming relaxation was proposed by Potts [10]. As observed in [6], this relaxation satisfies all the inequalities of the relaxation CT when deriving the completion times values from (6). Thus the optimal solution can be rounded as before to produce a 2-approximate solution.

2.3. The new relaxation

We replace the triangle inequalities (5) with the inequalities

$$\delta_{ki} \leqslant \delta_{kj} \quad \text{if } i \prec j, \;\; k \neq j, \;\; k \neq i. \tag{8}$$

These inequalities correspond in general to a *proper* subset of the triangle inequalities that guarantees that whenever $i \prec j$, the corresponding fractional completion times satisfy constraint (2).

The only variables δ_{ij} whose values are undetermined are those for which the jobs i and j are *unrelated*, which we denote $i\langle\;\rangle j$. Thus we write the new integer programming relaxation ISLO (*integer simplified linear ordering*) as follows:

$$(\mathsf{ISLO}) \quad \begin{aligned} \text{Min} \quad & C + \sum_{k\langle\;\rangle j} \delta_{kj} p_k w_j, \\ \text{s.t.} \quad & \delta_{kj} + \delta_{jk} = 1 \quad \text{for all } k\langle\;\rangle j, && (9) \\ & \delta_{kj} - \delta_{ki} \geqslant 0 \quad \text{if } i \prec j \text{ and } k\langle\;\rangle i,\; k\langle\;\rangle j, && (10) \\ & \delta_{kj} \in \{0, 1\} \quad \text{for all } k\langle\;\rangle j, && (11) \end{aligned}$$

where $C = \sum_{i \prec j} p_i w_j$. Note that since a subset of triangle inequalities (5) has been dropped, the above integer program is only a *relaxation* of the problem. Let SLO be the linear programming relaxation of ILO, where the integrality constraints (11) are replaced by $\delta_{kj} \geqslant 0$, $k\langle\;\rangle j$.

The following lemma establishes that the linear program SLO is stronger than the linear program CT. For notational simplicity, assume that we have defined all the δ_{ij}'s for $i \neq j$ (all the missing ones have their values determined by the precedence relations, and their contribution to the objective function appears in the constant C above). A similar lemma for LO was presented in [12]. For completeness we provide a full proof below.

Lemma 2.1. *Let* $\{\delta_{ij}\}$ *be a feasible solution to the linear program* SLO, *and define* $C_j := p_j + \sum_{k \neq j} \delta_{kj} p_k$, $j = 1, \dots, n$. *Then* $\{C_j\}$ *is a feasible solution to* CT.

Proof. To verify (1), fix any subset $S \subseteq N$, then

$$\begin{aligned} \sum_{j \in S} p_j C_j &= \sum_{j \in S} p_j \left(p_j + \sum_{k \neq j} \delta_{kj} p_k \right) \\ &= p^2(S) + \sum_{\substack{j \in S, k \in N \\ j \neq k}} \delta_{kj} p_j p_k \\ &\geqslant p^2(S) + \sum_{\substack{j, k \in S \\ j \neq k}} \delta_{kj} p_j p_k \\ &= p^2(S) + \sum_{\substack{j, k \in S \\ j < k}} (\delta_{kj} + \delta_{jk}) p_j p_k \\ &= \tfrac{1}{2}(p^2(S) + p(S)^2), \end{aligned}$$

where the last equality follows from (9).

To verify (2), suppose that $i \prec j$, then since $\delta_{ij} = 1$,

$$C_j = p_j + p_i + \sum_{k \neq j,i} \delta_{kj} p_k,$$

applying (10),

$$C_j \geqslant p_j + p_i + \sum_{k \neq j,i} \delta_{ki} p_k.$$

But since $\delta_{ji} = 0$,

$$p_i + \sum_{k \neq j,i} \delta_{ki} p_k = C_i,$$

so that (2) follows. □

A consequence of the lemma is that the value of the solution of SLO is within a factor of 2 from optimal. In particular, an optimal solution to SLO can be rounded to a 2-approximate schedule as in Section 2.1. Also if we denote $\mathsf{OPT_{LP}}$ the optimal objective value of the linear program LP, we have that

$$\mathsf{OPT_{CT}} \leqslant \mathsf{OPT_{SLO}} \leqslant \mathsf{OPT_{LO}}.$$

Examples from [6] show that the inequality on the left is tight. An interesting open question is whether the inequality on the right is tight or not. Indeed, it is not even clear whether the inequality is strict for the case of the integer programs ISLO and ILO.

Each constraint of ISLO has no more than two variables. Therefore, ISLO is a special case of IP2 studied by Hochbaum et al. [7],

$$\begin{array}{lll} & \text{Min} & \sum_{j=1}^{n} w_j x_j, \\ (\mathsf{IP2}) & \text{s.t.} & a_i x_{j_i} + b_i x_{k_i} \geqslant c_i \quad \text{for } i = 1,\ldots,m, \\ & & \ell_j \leqslant x_j \leqslant u_j \quad j = 1,\ldots,n, \\ & & x_j \quad \text{integer} \quad j = 1,\ldots,n, \end{array}$$

where $1 \leqslant j_i,\ k_i \leqslant n$, and all the coefficients are integer. Any IP2 problem that is feasible has a superoptimal half-integral solution derived in the time required to solve a minimum cut problem on a network with $O(nU)$ nodes and $O(mU)$ arcs, for $U = \max_{j=1,\ldots,n}(u_j - \ell_j)$, for n the number of variables and m the number of constraints. Moreover, the half-integral solution has a rounding of the components that are half-integer that is feasible and the resulting solution is 2-approximate for IP2.

For the integer program ISLO, with n the number of jobs, the network of Hochbaum et al. [7] has $O(n^2)$ nodes and $O(n^3)$ arcs. Furthermore, any IP2 problem has a superoptimal half-integral solution that is derived from the solution of the minimum cut problem on the respective network. In addition, there is an optimal solution that coincides with the half-integral solution on the integer components.

Consider now the linear program SLO. Since the integer program SLO has all constraint coefficients in

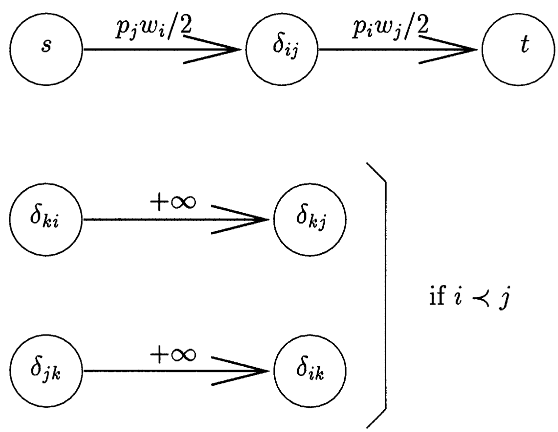

Fig. 1. Construction of the network $\mathcal{N}$.

$\{-1, 0, 1\}$, it follows from Lemma 6.1 of Hochbaum et al. [7] that the extreme points of the linear programming relaxation are half-integral. Namely, each basic feasible solution has each variable δ_{ij}, $i\langle\ \rangle j$, either 0, $\frac{1}{2}$ or 1. Indeed, as in [7], an optimal solution to the linear programming relaxation can be found via a minimum cut computation.

Note that the procedure of Hochbaum et al. [7] for generating a 2-approximation algorithm via rounding, is not applicable to SLO because rounding is not guaranteed to generate a feasible solution to LO. Instead it is necessary to compute the "fractional" completion times and thus derive a feasible sequence.

We construct here a specialized network for SLO, $\mathcal{N}$, as follows. The set of nodes consists of a source s and a sink t, and a node for each varaible δ_{ij}, $i\langle\ \rangle j$. There is an arc from s to node δ_{ij} with capacity $p_j w_i/2$, and an arc from δ_{ij} to t with capacity $p_i w_j/2$, for each pair $i\langle\ \rangle j$. Also, if $i \prec j$ there is an arc from δ_{ki} to δ_{kj} and one from δ_{jk} to δ_{ik}, each with infinite capacity (see Fig. 1). For any subset of nodes S, we will use $\bar{S}$ to denote the set of nodes not in S, and $(S, \bar{S})$ the set of arcs from S to $\bar{S}$, that is, the cut defined by the set S. The network $\mathcal{N}$ has $O(n^2)$ nodes and $O(n^3)$ arcs. The network just described is a simplification of the construction for IP2 of Hochbaum et al. [7].

Lemma 2.2. *Each feasible solution* $\{\delta_{ij}\}$ *to* ISLO *corresponds to a finite cut* $(S, \bar{S})$ *in* $\mathcal{N}$, *with* $s \in S$, $t \in \bar{S}$, *whose capacity is exactly the objective function value of the solution* $\{\delta_{ij}\}$.

Proof. Let S consist of the source s together with the nodes δ_{ij} for which the corresponding SLO value is 1. Suppose that $i \prec j$, and $k\langle\,\rangle i$, $k\langle\,\rangle j$. If $\delta_{ki} \in S$, $\delta_{ki}=1$, and then $\delta_{kj}=1$, so that $\delta_{kj}\in S$. Also, if $\delta_{jk}\in S$, $\delta_{jk}=1$, $\delta_{kj}=0$, so that $\delta_{ki}=0$ or $\delta_{ik}=1$ and then $\delta_{ik}\in S$. Thus the cut $(S,\bar{S})$ has finite capacity. Finally note that if $\delta_{ik}\in S$, $\delta_{ki}\notin S$ (because $\delta_{ik}=1$, so that $\delta_{ki}=0$), and the contribution of the pair $\{i,k\}$ to the capacity of the cut $(S,\bar{S})$ is $p_k w_i/2 + p_k w_i/2 = p_k w_i$, the same as the contribution of δ_{ik} and δ_{ki} to the objective function of SLO. □

Lemma 2.3. *Each finite cut $(S,\bar{S})$, with $s\in S$, $t\in\bar{S}$, corresponds to a half-integral solution $\{\delta_{ij}\}$, that is a feasible solution to the linear program* SLO *and whose value is precisely the capacity of the cut $(S,\bar{S})$.*

Proof. Set $\{\delta_{ij}\}$ as follows:

$$\delta_{ij} := \begin{cases} 1 & \text{if } \delta_{ij}\in S,\ \delta_{ji}\notin S,\\ 0 & \text{if } \delta_{ij}\notin S,\ \delta_{ji}\in S,\\ \frac{1}{2} & \text{if } \delta_{ij}\in S,\ \delta_{ji}\in S \text{ or } \delta_{ij}\notin S,\ \delta_{ji}\notin S.\end{cases}$$

A straightforward calculation shows that $\{\delta_{ij}\}$ satisfies the conditions of the lemma. □

Using Lemmas 2.2 and 2.3, we conclude the main result of our paper.

Theorem 2.4. *A half-integral solution to* SLO *can be found via a minimum cut computation in the network $\mathcal{N}$, whose objective function value is a lower bound on the optimal objective function value of* ISLO.

As a final remark, note that in the network $\mathcal{N}$ we can always send $\frac{1}{2}|p_i w_j - p_j w_i|$ units of flow directly from the source to the sink through node δ_{ij}, thus for the actual computation of the minimum cut, we can subtract from the capacity of the arcs (s,δ_{ij}) and (δ_{ij},t), the quantity $\frac{1}{2}\min(p_i w_j, p_j w_i)$ and eliminate the zero capacity arcs from the network. With this preprocessing, each δ_{ij} node is connected either to the source or the sink but not both.

3. A new 2-approximation algorithm for $1|\mathrm{prec}|\sum w_j C_j$

In the following theorem we generalize an observation made by Von Arnim et al. [1], for the special case in which all the weights are 1. Although the proof is straightforward, it has not been mentioned earlier in the literature.

Theorem 3.1. *For any instance of $1|\mathrm{prec}|\sum w_j C_j$, suppose that the weights and processing times are interchanged and the precedence relations are reversed. Then the new instance of $1|\mathrm{prec}|\sum w_j C_j$ is equivalent to the old one. More precisely, there is a one-to-one correspondence between feasible solutions that preserves costs.*

Proof. In the integer linear ordering formulation ILO, set $\bar{\delta}_{ik} := 1-\delta_{ik}$. Then $\{\bar{\delta}_{ik}\}$ is a feasible solution to the new instance, with the same objective function value. By symmetry, the result follows. Note that we have shown that the sequence of jobs $(a_1,\ldots,a_n)$ is feasible for the original instance if, and only, if the reversed sequence $(a_n,\ldots,a_1)$ is feasible for the new instance; in addition, both sequences have the same sum of weighted completion time values. □

Note that the proof of the theorem also establishes that the SLO relaxations of the two instances have the same objective function value.

The new approximation algorithm consists of applying the algorithm of Hall et al. [6], described in Section 2.1, to the new instance constructed as in the theorem — exchanging the roles of the weights and processing times. More precisely, we first solve the linear program SLO; let $\{\delta_{ij}\}$ be an optimal solution and let $\{\bar{\delta}_{ij}\}$ as in the proof of the theorem (i.e. $\bar{\delta}_{ij} = 1-\delta_{ij}$), so that $\{\bar{\delta}_{ij}\}$ is an optimal solution to the SLO linear program corresponding to the new instance in which processing times and weights have been interchanged, and the precedence graph has been reversed. We now construct the "fractional" completion times $T_j := w_j + \sum_k \bar{\delta}_{kj} w_k = w_j + \sum_k \delta_{jk} w_k$, $j=1,\ldots,n$, and assume without loss of generality that $T_1 \geqslant \cdots \geqslant T_n$, then the ordering $\{n,\ldots,1\}$ is feasible and within a factor of 2 from optimal in the new instance or, equivalently, the ordering $\{1,\ldots,n\}$ is feasible and within a factor of 2 from optimal in the original instance.

In effect, note that after solving the min-cut of Section 2.3, if $\{\delta_{ij}\}$ is an optimal solution, we obtain two 2-approximation algorithms: as in [6], use the sequence based on nondecreasing values of $C_j := p_j + \sum_k \delta_{kj} p_k$, $j=1,\ldots,n$, and as above use the sequence based on nonincreasing values of T_j. Thus at essentially the same cost in running time, we can produce two sequences that are guaranteed to be within a factor of 2 from optimal.

An interesting open question, addressed to us by Schulz [13] and an anonymous referee, is whether the two sequences are indeed different.

4. For further reading

[3,4,14,15]

Acknowledgements

We are grateful to David Shmoys for several helpful discussions. We also acknowledge Andreas Schulz for useful comments on an earlier version of the paper.

References

[1] A. Von Arnim, U. Faigle, R. Schrader, The permutahedron of series–parallel posets, Discrete Appl. Math. 28 (1990) 3–9.
[2] C. Chekuri, R. Motwani, Precedence constrained scheduling to minimize weighted completion time on a single machine, Discrete Appl. Math., to appear.
[3] D. Coppersmith, S. Winograd, Matrix computations via arithmetic progressions, Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, pp. 1–6.
[4] A.V. Goldberg, R.E. Tarjan, A new approach for the maximum flow problem, J. ACM 35 (1988) 921–940.
[5] R.L. Graham, E.L. Lawler, J.K. Lenstra, A.H.G. Rinnoy 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., to appear.
[7] D.S. Hochbaum, N. Meggido, J. Naor, A. Tamir, Tight bounds and 2-approximation algorithms for integer programs with two variables per inequality, Math. Programming 62 (1993) 69–83.
[8] E.L. Lawler, Sequencing jobs to minimize total weighted completion time subject to precedence constraints, Ann. Discrete Math. 2 (1978) 75–90.
[9] F. Margot, M. Queyranne, Y. Wang, Decompositions, network flows and a precedence constrained single machine problem, Working paper, 1997.
[10] C.N. Potts, An algorithm for the single machine sequencing problem with precedence constraints, Math. Programming Stud. 13 (1980) 78–87.
[11] M. Queyranne, Structure of a simple scheduling polyhedron, Math. Programming 58 (1993) 263–285.
[12] M. Queyranne, A.S. Schulz, Polyhedral approaches to machine scheduling, Report No 408/1994, Fachbereich Mathematik, Technische Universitt Berlin, 1994.
[13] A.S. Schulz, private communication, June 1997.
[14] P.M. Vaidya, A new algorithm for minimizing convex functions over convex sets, Proceedings of the 30th Annual IEEE Symposium on Foundations of Computer Science, 1989, pp. 338–343.
[15] P.M. Vaidya, Speeding-up linear programming using fast matrix multiplication, Proceedings of the 30th Annual IEEE Symposium on Foundations of Computer Science, 1989, pp. 332–337.
[16] L.A. Wolsey, Formulating single machine scheduling problems with precedence constraints, in: J.J. Gabsewicz, J.F. Richard, L.A. Wolsey (Eds.), Economic Decision Making: Games, Econometrics and Optimisation, Contributions in Honour of Jacques Dreze, North-Holland, Amsterdam, 1990, pp. 473–484.