Chin. Ann. Math.
26B:1(2005),153–158.

SEMI-DEFINITE RELAXATION ALGORITHM FOR SINGLE MACHINE SCHEDULING WITH CONTROLLABLE PROCESSING TIMES

CHEN Feng* ZHANG Liansheng**

Abstract

The authors present a semi-definite relaxation algorithm for the scheduling problem with controllable times on a single machine. Their approach shows how to relate this problem with the maximum vertex-cover problem with kernel constraints (MKVC). The established relationship enables to transfer the approximate solutions of MKVC into the approximate solutions for the scheduling problem. Then, they show how to obtain an integer approximate solution for MKVC based on the semi-definite relaxation and randomized rounding technique.

Keywords Scheduling with controllable times, Semi-definite programming, Approximation algorithm
2000 MR Subject Classification 90B35, 90C27, 68Q25

§ 1. Introduction

Scheduling problems with controllable times on a single machine can be stated as follows. Given a set of n jobs $J = \{1, 2, \cdots, n\}$. Each job $j \in J$ has a weight $W_j \in Z^+$ and a normal processing time $p_j \in Z^+$. The processing times of jobs are controllable in the following manner. The normal processing time of job j can be reduced by up to u_j units ($u_j \in z^+$ and $u_j \leq p_j$) if its processing is speeded up. Each unit reduction of processing time of job j requires a cost of c_j due to the fact that some additional resources are necessary for the speedup. In a given schedule , t_j denotes the reduction of processing time and $P_j = p_j - t_j$ the actual processing time of job j. Let C_j denote the completion time of job j. Our goal is to find a schedule of the jobs and a processing time reduction t_j ($t_j \leq u_j$) for each job j such that the total cost including the total weight completion time of jobs and the total cost of speedup, i.e., $\sum\limits_{j\in J} w_jC_j + \sum\limits_{j\in J} c_jt_j$, is minimum, which can be denoted as $1/cpt/\sum w_jC_j + c_jt_j$, where the notion "cpt" stands for "controllable processing times".

The problem $1/cpt/\sum w_jC_j + c_jt_j$ is NP-hard (see [1]). Vickson [2] shows that for the problem $1/cpt/\sum w_jC_j + c_jt_j$, there is an optimal schedule that satisfies the following all-or-one property: the processing time of each job $j \in J$ is either fully reduced or not reduced at all, i.e., $t_j \in \{0, u_j\}$, and its actual processing time $P_j \in \{p_j, p_j - u_j\}$. Huang and Zhang [3] give a polynomial time algorithm for $1/cpt/\sum w_jC_j + c_jt_j$ with $u_j \equiv u$ and $c_j \equiv c$.

Manuscript received September 11, 2002. Revised July 1, 2003.
*Corresponding author. **E-mail:** fchen@sjtu.edu.cn
Department of Mathematics, Shanghai University, Shanghai 200436, China.
Department of Industrial Engineering and Management, Shanghai Jiaotong University, Shanghai 200030, China.
**Department of Mathematics, Shanghai University, Shanghai 200436, China.

Since the seminar work of Goemans and Williamson [4] for MAX-CUT, semidefinite programming (SDP) relaxations have recently proved useful in obtaining improved approximation algorithms for several combinatorial problems, including MAX2SAT (see [4]), MAX k-CUT and MAX BISECTION (see [5]), non-convex quadratic (see [6]), quadratic programming (see [7]) and so on. A review is given in [8].

Semi-definite programming (SDP) can be stated as follows:

$$\begin{aligned} \min \quad & C \cdot X. \\ \text{s.t.} \quad & A_i \cdot X = b_i, \\ & X \succeq 0. \end{aligned}$$

where, C, A_i $(i = 1, \cdots, m)$ are $n \times n$ real symmetric matrices, and A_i $(i = 1, 2, \cdots, m)$ are linear independent, $b \in R^m$, $A \cdot B = \sum\limits_{i,j} A_{ij}B_{ij} = \mathrm{Tr}(A^T B)$ denotes the inner product of matrix A, B; $X \succeq 0$ represents X is a semi-definite matrix. Semi-definite programming can be solved by ellipsoid algorithm (see [9]) and interior-point polynomial-time methods (see [**?**]).

It is known that Skutella is the first researcher who has successfully used the convex and semidefinite programming techniques to design the approximation algorithms for scheduling. In [**?**], Skutella presents a $\frac{3}{2}$-approximation for $R//\sum w_j C_j$ and 1.2752-approximation for $R2//\sum w_j C_j$. Inspired by the paper, Zhang, Tang and Chen [**?**] give a 3/2 approximation algorithm for a general problem based on convex quadratic programming relaxation, $1/cpt/\sum w_j C_j + c_j t_j$, as a special case, can also be solved by the methods. Recently, Xu [**?**] also used the same technique occurring in [**?**] to get an improvement algorithm to 1.27-approximation for $1/cpt/\sum w_j C_j + c_j t_j$.

Although it is difficult for us to solve minimum combinatorial optimization problems directly by the technique of semi-definite programming relaxation, Skutella [**?**] gets a 1.122-approximation for $P_2//\sum w_j C_j$ by translating the problem $P_2//\sum w_j C_j$ into a Max cut problem. In this paper, we use the similar idea to develop a semi-definite relaxation algorithm for $1/cpt/\sum w_j C_j + c_j t_j$.

This paper is organized as follows. In Section 2 we show that the problem $1/cpt/\sum w_j C_j + c_j t_j$ is equivalent to a maximum vertex-cover problem with kernel constraints (MKVC). In Section 3, we point out that the algorithm for (MKVC) can be employed to solve the problem $1/cpt/\sum w_j C_j + c_j t_j$ and has a better performance under some reasonable assumptions.

§ 2. Equivalence of Two Problems

In the section, we will show that the problem $1/cpt/\sum w_j C_j + c_j t_j$ is equivalent to the maximum vertex-cover problem with kernel constraints which can be stated as follows. Let $G = (V, E)$ be an undirected graph where $V = \{1, 2, \cdots, n\}$ is the set of vertexes, E is the set of edges. Given kernel sets $K_1, K_2, \cdots, K_m \subset V$, each kernel set has just two elements. There are weights $w_{ij} = w_{ji} \in Z^+$ on edge $(i, j) \in E$ and each vertex has a weight $w_i \in Z^+$. The problem is to determine a subset $S \subset V$ such that the sum of total weight of the edges covered by S and the total weight of the vertex in S is maximized with kernel constraints, i.e., just one element of K_i will be covered by S. We will denote this problem by MKVC. Without loss of generality, we assume that no two kernel sets have common element, otherwise there is not a feasible solution for MKVC. Let

$$x_i = \begin{cases} 1, & \text{if } \ i \in S, \\ 0, & \text{if } \ i \notin S. \end{cases}$$

Then the problem MKVC can be formulated as follows:

$$\begin{aligned} \text{MKVC}: \quad & \max_{S\subset V} \sum_{i \text{ or } j\in S,\ i<j} w_{ij} + \sum_{i\in S} w_i \\ & \text{s.t.} \quad x_e + x_f = 1, \qquad e,f \in K_i, \qquad i = 1,\cdots,m, \\ & \qquad\qquad\qquad\quad x_j \in \{0,1\}, \quad j = 1,\cdots,n. \end{aligned}$$

Based on the property of all-or-none, we can see that the $1/cpt/\sum w_jC_j + c_jt_j$ is equivalent to the following problem with non-controllable processing times. Given a set of $2n$ jobs $H = \{1,2,\cdots,2n\}$. Each job $j \in H$ has a weight v_j and a non-controllable processing time T_j if it is processed on machine, where $v_j = w_j$, $T_j = p_j - \mu_j$ if $j \le n$ and $v_j = w_{j-n}$, $T_j = p_{j-n}$ if $j > n$. If job $j \in H$ is processed on the single machine, it incurs a processing cost d_j with $d_j = c_j\mu_j$ if $j \le n$ and $d_j = 0$ if $j > n$. The problem is to select a subset of n jobs $K \subset H$ and find a schedule for these jobs such that, for each $j = 1,\cdots,n$, exactly one of the two jobs $\{j, n+j\}$ is selected, and the sum of the total weight completion time and the total processing cost of these jobs is minimum. We will focus on this equivalent problem in the remainder of the paper.

We will show how an approximation algorithm for MKVC translates into an approximation algorithm for $1/cpt/\sum w_jC_j + c_jt_j$. Firstly, we associate with each instance of the $1/cpt/\sum w_jC_j + c_jt_j$ problem a complete undirected $G = (V,E)$ in the following way: the vertex set $V = \{1,2,\cdots,2n\}$ corresponds to the job set $\{J_1, J_2, \cdots, J_{2n}\}$; and the weight w_{ij} of the edge $(i,j) \in E$ is given by

$$w_{ij} = \min\{v_iT_j, T_iv_j\} \qquad \text{for } \ i \ne j.$$

The reason for this setting is, for any single machine scheduling problem with our objective function, the optimal schedule will assign the jobs with the order of nondecreasing T_j/v_j. Moreover, let the weight w_i of vertex i be $v_iT_i+d_i$. Given kernels $K_i = \{i, n+i\}$, $i = 1,\cdots,n$, we have constructed an instance of MKVC.

It is easy to see that any feasible schedule can be interpreted as a partition $(S, V\backslash S)$, where S corresponds to the constraints that exactly one of the two jobs $\{j, n+j\}$ is selected. Moreover, the total weighted completion time of a feasible schedule can be represented by the total weights of the edges with both endpoints in subset S plus a term $\sum\limits_{j\in S} v_jT_j + d_j$. One can verify

$$c(E) + \sum_{j\in S}(v_jT_j + d_j) = \sum_{j\in S}(v_jC_j + d_j) + W(V\backslash S) - \sum_{j\in V\backslash S}(v_jT_j + d_j), \tag{2.1}$$

where $c(E) = \sum\limits_{1\le i<j\le n} w_{ij}$ is the total weights of all edges of E; C_j is the completion time of job J_j in the optimal schedule corresponding to the set S; and $W(V\backslash S)$ denotes the objective function value corresponding to $V\backslash S$ for the above instance of MKVC, i.e.,

$$W(V\backslash S) = \sum_{i \text{ or } j\in V\backslash S} w_{ij} + \sum_{j\in V\backslash S}(v_jT_j + d_j).$$

Furthermore, the equation (**??**) can be reformulated as

$$c(E) + \sum_{j=1}^{n}(v_jT_j + d_j) = \sum_{j\in S}(v_jC_j + d_j) + W(V\backslash S). \tag{2.2}$$

Note that, for a given instance of the $1/cpt/\sum w_jC_j + c_jt_j$, $c(E) + \sum\limits_{1\le j\le 2n}(v_jT_j + d_j)$ is a constant. Therefore, minimizing the total weighted completion time $\sum\limits_{j\in S}(v_jC_j + d_j)$ of the schedule is equivalent to maximizing the vertex cover value of $W(V\backslash S)$. This illustrate that any algorithm for MKVC can be employed to solve a minimum problem $1/cpt/\sum w_jC_j+c_jt_j$. The minimal value of $1/cpt/\sum w_jC_j+c_jt_j$ problem is denoted as Z^* and the maximal value of the corresponding $W(V\backslash S)$ is denoted as W^*.

From the equivalence of the above two problems, we can claim that MKVC is NP-hard, because of the NP-hardness of $1/cpt/\sum w_jC_j + c_jt_j$.

§ 3. Approximation Algorithm

We should note that the problem MKVC presented in this paper differs from the traditional Max-Vertex-Cover (MVC) models, which was introduced by Petrank [**?**]. Although no results on this problem have been reported in the literature, it is easily to know that MKVC can be reduced to the MAX 2-SAT problem. Since that, we can also employ the algorithms for MAX 2-SAT to solve MKVC with the same approximation ratio. Based on semi-definite relaxation, Shiro Matuura1 and Tomomi Matsui [**?**] refined the rotation technique and proposed a 0.935-approximation for MAX 2-SAT which beats the 0.9310900680 approximation in [**?**]. For the reasons of brevity, here we omit the details of the algorithm. For convenience, we denote the algorithm occurring in [**?**] as (SDP).

Let

$$k = \frac{c(E) + \sum\limits_{1\le j\le 2n}(V_jT_j + d_j)}{W^*}.$$

We have

Theorem 3.1. *The expect performance ratio of the semi-definite programming relaxation algorithm (SDP) is at least* $1 + \frac{0.065}{k-1}$ *for* $1/cpt/\sum w_jC_j + c_jt_j$.

Proof. Let $W(V\backslash\bar{S})$ corresponds to the solution by (SDP) such that $E(W(V\backslash\bar{S})) \ge \frac{k-0.935}{k-1}W^*$. By (**??**), we can see that

$$\sum_{j\in S}(v_jC_j + d_j) + W(V\backslash\bar{S}) = Z^* + W^*,$$

then

$$\begin{aligned}
E\Big(\sum_{j\in\overline{S}}(v_jC_j + d_j)\Big) &= Z^* + W^* - E(W(V\backslash\overline{S})) \\
&\le Z^* + W^* - 0.935W^* \\
&\le Z^* + 0.065W^* \\
&\le \Big(1 + 0.065\frac{W^*}{Z^*}\Big)Z^* \\
&\le \left(1 + 0.065\frac{W^*}{c(E) + \sum\limits_{1\le j\le 2n}(V_jT_j + d_j) - W^*}\right)Z^* \\
&= \Big(1 + 0.065\frac{1}{k-1}\Big)Z^*.
\end{aligned}$$

This completes the proof.

Let $W_p^L = \min\{w_i p_j : i, j = 1, \cdots, n\}$, $W_p^U = \max\{w_i p_j : i, j = 1, \cdots, n\}$ and $r_s = \frac{W_p^U}{W_P^L}$.

Theorem 3.2. *If $r_s \leq 1.29$, then the expect performance ratio of algorithm SM for $1/cpt/\sum w_j C_j + c_j t_j$ is at least 1.27 for sufficiently large n.*

Proof. Note that

$$\begin{aligned}\frac{Z^*}{W^*} &\geq \frac{C_n^2 W_p^L}{C_{2n}^2 W_p^U - C_n^2 W_p^L} \\ &= \frac{n(n-1)W_p^L}{2n(2n-1)W_p^U - n(n-1)W_p^L} \\ &= \frac{n^2 - n}{(4r_s - 1)n^2 - (2r_s - 1)n}.\end{aligned}$$

As n increases, $\frac{n^2-n}{(4r_s-1)n^2-(2r_s-1)n}$ approaches $\frac{1}{4r_s-1}$ and therefore

$$\frac{Z^*}{W^*} \geq \frac{1}{4r_s - 1}.$$

Moreover

$$\begin{aligned}k &= \frac{c(E) + \sum\limits_{1 \leq j \leq 2n} (V_j T_j + d_j)}{W^*} \\ &= \frac{Z^* + W^*}{W^*} = 1 + \frac{Z^*}{W^*} \\ &\geq 1 + \frac{1}{4r_s - 1}.\end{aligned}$$

By simple calculation, Theorem **??** gives the desired result.

It should be noted that the above randomized algorithm (SDP) (see [**?**]) can be derandomized in polynomial time following the method of conditional probabilities, see [**?**] for details.

From the proof of the theorem, we can see that the expect approximation ratio of the algorithm (SDP) for $1/cpt/\sum w_j C_j + c_j t_j$ is related to the parameter k. As k is increased, we can obtain better algorithm for the problem $1/cpt/\sum w_j C_j + c_j t_j$. Though the lower bound of k in the theorem is likely to be true for most instances, we can not determine the lower bound of k exactly. In fact, we try to give the bound of k depending on the bound given by the convex quadratic relaxation of $1/cpt/\sum w_j C_j + c_j t_j$. It is a pity that no satisfying constant bound can be found. On the other hand, it is likely that we can not get a satisfying bound because k may become very closed to 1 with the value of d increasing to very large. However, if we assume that the $\{w_i p_j | i, j = 1, \cdots, n\}$ values are not too disparate, i.e., r_s is not too large, the algorithm (SDP) is guaranteed to have the worst performance ratio better than 1.27. We strongly believe that the algorithm (SDP) will performance better for most instances of single machine scheduling with controllable processing times.

Acknowledgement. The authors would like to thank Professor Yao Enyu and Tang Guochun for useful discussions on these subjects.

References

[1] Hoogeveen, J. A. & Woeginger, G. J., Scheduling with controllable processing times, Department of Mathematics, TU Graz, Graz, Austria.

[2] Vickson, R. G., Choosing the job sequence and processing times to minimize total processing plus flow cost on single machine, *Oper. Res.,* **28**(1980), 1155–1167.

[3] Huang, W. & Zhang, F., An $o(n^2)$ algorithm for a controllable machine scheduling problem, *IMA J. Math. Appl. Business Industry,* **10**(1999), 15–26.

[4] Goemans, M. X. & Williamson, D. P., Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming, *Journal of the Association for Computing Machinery,* **42**(1995), 1115–1145.

[5] Frieze, A. & Jerrum, M., Improved approximation algorithms for max k-cut and max bisection, *Algorithmica,* **18**(1997), 67–81.

[6] Nesterov, Yu. E., Semidefinite relaxation and nonconvex quadratic optimization, *Optimization Methods and Software*, **9**(1998), 141–160.

[7] Ye, Y. Y., Approximating quadratic programming with bound and quadratic constraints, *Mathematical Programming,* **84**(1999), 219–226.

[8] Goemans, M. X., Semidefinite programming in combinatorial optimization, *Mathematical Programming,* **79**(1997), 143–161.

[9] Grötschel, M., Lovász, L. & Schrijver, A., Geometric algorithms and combinatorial optimization, in Algorithms and Combinatorics, Vol. 2, Springer, Berlin, 1988.

[10] Alizadeh, F., Interior point methods in semidefinite programming with applications to combinatorial optimization, *SIAM Journal on Optimization,* **5**(1995), 13–51.

[11] Skutella, M., Semidefinite relaxation for parallel machine scheduling, in Proceedings of 39th Annual IEEE Symposium on Foundations of Computer Science, 1998, 472–481.

[12] Zhang, F., Tang, G. C. & Chen, Z. L., A 3/2-approximation algorithm for parallel machine scheduling with controllable processing times, *Operatins Research Letters,* **29**(2001), 41–47.

[13] Xu, D. C., An approximation algorithm for single machine scheduling with controllable processing times (in Chinese), *Acta Mathematica Sinica,* **46:**6(2003), 1047–1054.

[14] Petrank, E., The hardness of approximation: gap location, *Computational Complexity,* **4**(1994), 133–157.

[15] Shiro Matuura & Tomomi Matsui, 0.935-approximation randomized algorithm for max 2sat and its derandomization, http://www.keisu.t.u-tokyo.ac.jp/Research/METR/2001/METR01-03.pdf, September 2001.

[16] Uri Zwick, Analyzing the MAX 2-SAT and the MAX DI-CUT approximation algorithms of Feige and Goemans, Department of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel, March, 2000.

[17] Motwani, R. & Raghavan, P., Randomized Algorithms, Cambridge University Press, 1995.