# On the Budgeted MAX-CUT problem and its Application to the Capacitated Two-Parallel Machine Scheduling *

Jiawei Zhang and Yinyu Ye

March 13, 2001

**Keywords**: analysis of algorithms, Max-Cut, parallel machine scheduling.

## 1 Introduction

In the Budgeted MAX-CUT (BMAX-CUT) problem, we are given an undirected graph $G = (V, E)$ with the node set $V = \{1, 2, \cdots, n\}$, non-negative edge-weights $w_{ij} = w_{ji}$ on edges $(i, j) \in E$, and positive node-weights $n_i$ on nodes $i \in V$. The problem is to find a subset of nodes $S \subset V$ that maximizes

$$cut(S, V \setminus S) := \sum_{(i,j)\in E, i\in S, j\in V\setminus S} w_{ij}$$

subject to $\sum_{i\in S} n_i \leq T$ for a given $T$, where $0 \leq T \leq \frac{1}{2}\sum_{i\in V} n_i$.

There are several approximation algorithms for a problem related to BMAX-CUT in which each node has unit weight, i.e, $n_i = 1$ for $i = 1, 2, \cdots, n$, and the goal is to find a subset $S$ with $|S| = T$ such that $cut(S, V \setminus S)$ is maximized. Ageev and Sviridenko [1] designed a deterministic $\frac{1}{2}$-approximation algorithm for the unit weight problem using linear programming; Hassin and

*Department of Management Sciences, Henry B. Tippie College of Business, The University of Iowa, Iowa City, Iowa 52242, USA; and the second author is currently visiting the Department of Systems Engineering and Engineering Management, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong. Research supported in part by NSF grants DMI-9908077 and DMS-9703490. Research supported in part by NSF grants DMI-9908077 and DMS-9703490.

Rubinstein [6] obtain the same result using Bar-Yehuda's algorithm for partial vertex cover [2]. Zwick and Halperin [4] claimed a 0.701-approximation algorithm for the case $T = n/2$, which is improved from the previous results 0.699 (Ye [16]) and 0.651 (Frieze and Jeurrum [3]). Han, Ye and Zhang [5] extended the result to general $T$. All these results are obtained by solving a semidefinite programming relaxation.

However, all the algorithms mentioned above do not apply to the general BMAX-CUT problem. In this paper, we give two approximation algorithms which have performance guarantees 3/8 and 0.225, one solve linear programming relaxations and the other uses a simple greedy method. Our algorithms are based on the approximation algorithms for the so-called budgeted maximum coverage (BMCG) problem. In solving the BMCG problem, the goal is to find a subset $S$ with $\sum_{i\in S} n_i \le T$ such that the total weight of edges, which has at least one node in $S$, is maximized [8].

As an application of these two results, we consider the capacitated two-parallel machine scheduling problem, where we are given a set of $n$ jobs that have to be scheduled on two identical parallel machines. Each job $J_j$ $(j = 1, 2, \cdots, n)$ has a positive processing time $p_j$ and a positive weight $w_j$. In a feasible schedule, every job $J_j$ is processed for $p_j$ time units on one of the two machines without interruption. Each machine can process at most one job at a time, and one of the machines is available for a specified period of time $[0, T]$, where $T > 0$, after which it cannot process any job. The objective is to minimize the sum of weighted job completion times $\sum_{1\le j\le n} w_j C_j$, where $C_j$ is the completion time of job $j$ in a given schedule. We will denote this problem by $P2/T/\sum w_j C_j$. This problem has been studied by Kao and Elsayed [7], Lee and Liman [11], and Lee [9]. The unweighted problem, i.e, $w_j = 1$ for all $j$, is $NP-hard$ and a modified Shortest Processing Time (SPT) algorithm has performance guarantee $\frac{3}{2}$ due to Lee and Liman [11]. For more results on the capacitated parallel machine scheduling, see the surveys by Lee, Lei, and Pinedo [10], and Sanlaville and Schmidt [13].

We associate with each instance of the $P2/T/\sum w_j C_j$ problem a complete undirected graph $G = (V, E)$ in the following way: the node set $V = \{1, 2, \cdots, n\}$ corresponds to the job set $\{J_1, J_2, \cdots, J_n\}$; the weight $w_{ij}$ of the edge $(i, j) \in E$ is given by $w_{ij} = \min\{w_i p_j, p_i w_j\}$; the weight $n_i$ of the node $i \in V$ is given by $n_i = p_i$. 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 $p_j/w_j$ [12]. Then, the partition $(S, V \setminus S)$ of the node set $V$ can be interpreted as scheduling the $n = |V|$ jobs on the two machines. And $\sum_{i \in S} n_i \leq T$ corresponds to the availability constraint that one of the machines is available only in period $[0, T]$. Moreover, the total weighted completion time of a feasible schedule can be represented by the total weights of the edges with both nodes in the same subset $S$ or $V \setminus S$ plus a constant term $\sum_{1 \leq j \leq n} w_j p_j$. One can verify

$$w^{TO} + \sum_{1 \leq j \leq n} w_j p_j = \sum_{1 \leq j \leq n} w_j C_j + cut(S, V \setminus S), \tag{1}$$

where $w^{TO} = \sum_{1 \leq i < j \leq n} w_{ij}$ is the total weights of all the edges of $E$; $C_j$ is the completion time of job $J_j$ in the schedule corresponding to the partition $(S, V \setminus S)$. Note that, for a given instance of the $P2/T/\sum w_j C_j$ problem, $w^{TO} + \sum_{1 \leq j \leq n} w_j p_j$ is a constant. Therefore, by (1), minimizing the total weighted completion time $\sum_{1 \leq j \leq n} w_j C_j$ of the schedule is equivalent to maximizing the value of $cut(S, V \setminus S)$. This idea was first proposed by Skutella [14] for $P2//\sum w_j C_j$, i.e., the two-parallel machine scheduling problem without the capacity constraint; also see Yang, Ye and Zhang [15]. The following theorem is essentially the same as Theorem 5.1 in [14].

**Theorem 1** *For any $\rho \leq 1$, a $\rho$-approximation algorithm for the BMAX-CUT problem can be translated into an approximation algorithm for $P2/T/\sum w_j C_j$ with a performance guarantee at most $2 - \rho$.*

Therefore, the approximation algorithms for BMAX-CUT problem with performance guarantees $\frac{3}{8}$ and 0.225 implies the approximation algorithms for $P2/T/\sum w_j C_j$ with performance guarantees $\frac{13}{8}$ and 1.775. The rest of this paper is devoted to present the approximation algorithms for BMAX-CUT problem.

## 2 Approximation Algorithm for BMAX-CUT

For any $S \in V$, let $w(S)$ be the total weight of edges covered by $S$, $I(S)$ be the total weight of edges whose both nodes are in $S$. Then, we have $w(S) = I(S) + cut(S, V \setminus S)$. Also, let $n(S) = \sum_{i \in S} n_i$, the total node-weight of $S$.

**Lemma 1** *For any $0 \le \rho \le 1$, we can obtain a $\frac{\rho}{2}$-approximation solution for BMAX-CUT from a $\rho$-approximation solution of BMCG in $O(n^2)$ times.*

**Proof.** Note that the optimal value of BMAX-CUT, $w^*$, is bounded above by the maximal value of BMCG. Suppose $S$ is a $\rho$-approximation solution for BMCG, then $w(S) \ge \rho w^*$.

It is well known that one can find a partition of $S = (S^1, S^2)$ such that $cut(S^1, S^2) \ge \frac{1}{2} I(S)$ by several well-known $1/2-$approximation algorithms for max-cut problem in $O(n^2)$ times (e.g., simple randomization, greedy algorithm, local search). Note that

$$w(S) = I(S) + cut(S, V \setminus S) = I(S) + cut(S^1, V \setminus S) + cut(S^2, V \setminus S);$$

and

$$cut(S^i, V \setminus S^i) = cut(S^i, V \setminus S) + cut(S^1, S^2), \quad i = 1, 2.$$

Therefore

$$\begin{aligned} & cut(S^1, V \setminus S^1) + cut(S^2, V \setminus S^2) \\ = \;& 2 \cdot cut(S^1, S^2) + cut(S^1, V \setminus S) + cut(S^2, V \setminus S) \\ \ge \;& I(S) + cut(S, V \setminus S) = w(S) \ge \rho w^*. \end{aligned}$$

It follows that either $cut(S^1, V \setminus S^1) \ge \frac{\rho}{2} w^*$ or $cut(S^2, V \setminus S^2) \ge \frac{\rho}{2} w^*$ holds. Furthermore, since $S$ is a feasible solution for (BMCG), $\max\{n(S^1), n(S^2)\} \le n(S) \le T$. Thus either $S^1$ or $S^2$ is the $\frac{\rho}{2}-$approximation solution for BMAX-CUT. ■

Ageev and Sviridenko [1] developed a $\frac{3}{4}$-approximation algorithm for BMCG by solving $O(n^4)$ times linear programming relaxations. Khuller, Moss and Naor [8] presented a combinatorial algorithm for BMCG with performance guarantee $1 - \frac{1}{e}$ and running time $O(n^4 \log n)$. They also showed that a simple $O(n^2)$ greedy method had a performance guarantee $\frac{1}{2} - \frac{1}{2e}$. These results, combined with Lemma 1, lead to the following:

**Theorem 2** *For BMAX-CUT problem, There exist a $\frac{3}{8} = 0.375$-approximation algorithm by solving $O(n^4)$ times linear programming relaxations; A $(\frac{1}{2} - \frac{1}{2e}) = 0.316$-approximation algorithm with running time $O(n^4 \log n)$; A $(\frac{1}{4} - \frac{1}{4e}) = 0.158$-approximation algorithm with running time $O(n^2)$.*

## 3 Improved Analysis of the Simple Greedy Method

In this section, we show that a small modification to the simple $O(n^2)$ greedy method of Khuller, Moss and Naor [8] for BMCG leads to a 0.225-approximation algorithm for BMAX-CUT, which is improved from the original 0.158.

We first describe the

**Greedy Algorithm**

**Step 1. Set** $S = \emptyset, U = \emptyset, N = 0, G = V$.

**Step 2. If** $G = \emptyset$**, stop; otherwise, select** $i \in G$ **that maximizes** $\frac{w'(\{i\})}{n_i}$**, where** $w'(\{i\})$ **is the total weight of edges covered by** $i$ **but not by** $S$**.**

**if** $n_i > T$ **then** $G = G \setminus \{i\}$ **and go to step 2;**

**else if** $N + n_i \leq T$**,**

$S = S \cup \{i\}$**,** $N = N + n_i$**,** $G = G \setminus \{i\}$**and go to step 2;**

**else** $U = \{i\}$ **and stop;**

**end if**

**end if.**

**Step 3. Apply a simple Max-Cut algorithm to find a partition** $(S^1, S^2)$ **of** $S$ **such that** $cut(S^1, S^2) \geq \frac{1}{2}I(S)$**, and output the best subset among** $S^1$**,** $S^2$ **and** $U$**.**

The simple Greedy Algorithm runs $O(n^2)$ times. Now, we analyze the performance guarantee of the algorithm. Let $S^{OPT}$ denote an optimal node-subset of BMCG.

**Lemma 2** *If* $U = \emptyset$*, then either* $S^1$ *or* $S^2$ *is a* $\frac{1}{2}$*-approximation solution for BMAX-CUT.*

**Proof.** Note that $U = \emptyset$ implies that all the nodes $i$ with $n_i \leq T$ has been added to $S$. Thus, $S^{OPT} \subset S$ and $S$ is optimal for BMCG. By Lemma 1, either $S^1$ or $S^2$ is a $\frac{1}{2}$-approximation solution for BMAX-CUT. ■

At the rest of this section, we assume that $U \neq \emptyset$. Let $l$ be the total number of nodes added to $S$ before the $(l+1)$th node in $U$ is added. Let $S_i,\ i = 1, 2, \cdots, l$ denote the first $i$ nodes added in $S$, and $S_{l+1} = S_l \cup U$. The following lemma is due to Khuller, Moss and Naor [8].

**Lemma 3** *For $i = 1, 2, \cdots, l+1$, the following holds:*

$$w(S_i) \geq \left[1 - \prod_{k=1}^{i}(1 - \frac{n_k}{T})\right] \cdot w(S^{OPT})$$

**Sketch of Proof.** First they prove that

$$w'(\{i\}) = w(S_i) - w(S_{i-1}) \geq \frac{n_i}{T}(w(S^{OPT}) - w(S_{i-1})).$$

Notice that $w(S^{OPT}) - w(S_{i-1})$ is the total weight of the edges covered by $S^{OPT}$ but not by $S_{i-1}$. And for each nodes $k$ in $S^{OPT} \setminus S_{i-1}$, the ratio $\frac{w'(\{k\})}{n_k} \leq \frac{w'(\{i\})}{n_i}$. Furthermore, the total node-weight in $S^{OPT} \setminus S_{i-1}$ is at most $T$. Then, one can prove the lemma by induction. ■

**Lemma 4** *The function $f(x) = e^{-x}x$ is increasing in $[0, 1]$.*

**Proof.** Notice that $f'(x) = (1-x)e^{-x} \geq 0$ if $x \in [0, 1]$. ■

**Lemma 5** *For $a_1, a_2, \cdot, a_n \in R^+$ such that $\sum_{i=1}^{n} a_i = A$, and $B \geq \max\{a_i,\ i = 1, ..., n\}$; we have $\prod_{i=1}^{n}(1 - \frac{a_i}{B}) \leq e^{-\frac{A}{B}}$.*

**Proof.** $\prod_{i=1}^{n}(1 - \frac{a_i}{B}) \leq (1 - \frac{A}{nB})^n = \left((1 - \frac{1}{n\frac{B}{A}})^{n\frac{B}{A}}\right)^{\frac{A}{B}} \leq e^{-\frac{A}{B}}$. ■

**Theorem 3** *The Greedy Algorithm has performance guarantee at least 0.225 for BMAX-CUT.*

**Proof.** Let $x = \frac{\sum_{i=1}^{l} n_i}{T}$ and $y = \frac{n_{l+1}}{T}$. Then, we have $0 < x \leq 1$, $0 < y \leq 1$, and $x + y > 1$. By Lemma 3 and Lemma 5, we have

$$\begin{aligned}
\frac{w(S_{l+1})}{w(S^{OPT})} &\geq 1 - \prod_{k=1}^{l+1}(1 - \frac{n_k}{T}) \\
&= 1 - \prod_{k=1}^{l}(1 - \frac{n_k}{T}) \cdot (1 - \frac{n_{l+1}}{T}) \\
&\geq 1 - e^{-x} \cdot (1 - y) \\
&\geq 1 - e^{-x}x = 1 - f(x) \geq 1 - \frac{1}{e}.
\end{aligned}$$

Note that $w(U) \geq w^{'}(U) := w(S_{l+1}) - w(S_l)$. Thus, If $w^{'}(U) \geq \frac{1}{3}(1-\frac{1}{e}) \cdot w(S^{OPT})$, then we get

$$cut(U, V \setminus U) = w(U) \geq w^{'}(U) \geq \frac{1}{3}(1-\frac{1}{e}) \cdot w(S^{OPT}) \geq \frac{1}{3}(1-\frac{1}{e}) \cdot w^*.$$

Otherwise, $w(S) = w(S_l) \geq \frac{2}{3}(1-\frac{1}{e}) \cdot w(S^{OPT})$. Then by Lemma 1, either $S^1$ or $S^2$ is $\frac{1}{3}(1-\frac{1}{e}) = 0.2107$ approximation solution for BMAX-CUT.

However, a more careful analysis can improve the bound further. Let

$$t = \frac{e^{-1}}{1-e^{-1}} \quad \text{and} \quad \alpha = \frac{2-\sqrt{1+2t}}{3-2t}.$$

Then, if $\frac{w^{'}(U)}{w(S_{l+1})} \leq \alpha$, we have

$$w(S) = w(S_l) \geq (1-\alpha)(1-e^{-1})w(S^{OPT}),$$

which again implies that

$$\max\{cut(S^1, V \setminus S^1), cut(S^2, V \setminus S^2)\} \geq \frac{1}{2}(1-\alpha)(1-e^{-1})w(S^{OPT}). \tag{2}$$

On the other hand, if $\frac{w^{'}(U)}{w(S_{l+1})} \geq \alpha$, then we must also have

$$\frac{y}{x+y} = \frac{n_{l+1}}{\sum_{i=1}^{l+1} n_i} \geq \frac{w'(\{l+1\})}{\sum_{i=1}^{l+1} w'(\{i\})} = \frac{w^{'}(U)}{w(S_{l+1})} \geq \alpha,$$

since the maximizing-$\frac{w^{'}(\{i\})}{n_i}$ rule of selecting the next node in $G$. Therefore,

$$y \geq \frac{\alpha}{1-\alpha}x.$$

By lemma 4 and the analysis above, we have

$$\frac{w(S_{l+1})}{w(S^{OPT})} \geq 1 - e^{-x}(1-y) \geq 1 - \frac{1-2\alpha}{1-\alpha}e^{-x}x \geq 1 - \frac{1-2\alpha}{1-\alpha}e^{-1}.$$

By the assumption, we have

$$cut(U,\ V \setminus U) = w(U) \geq w^{'}(U) \geq \alpha w(S_{l+1}) \geq \alpha\left(1 - \frac{1-2\alpha}{1-\alpha}e^{-1}\right)w(S^{OPT}). \tag{3}$$

Note that the selection of $\alpha$ makes the right-hand sides of both (2) and (3) equal, i.e.,

$$\frac{1}{2}(1-\alpha)(1-e^{-1}) = \alpha\left(1 - \frac{1-2\alpha}{1-\alpha}e^{-1}\right) > 0.225.$$

Thus, the best cut of $S^1$, $S^2$ and $U$ must be greater than $0.255w^*$. ■

A by-product of our analysis is that the simple greedy algorithm has a performance guarantee at least $\frac{1}{2} - \frac{1}{2e^2} = 0.432$ for BMAX-CUT if all nodes have unit cost 1. First, we run the greedy algorithm until $|S| = 2T$ where $T$ is the number of nodes we want to choose. By Lemma 3 and Lemma 5 we have

$$\frac{w(S)}{w(S^{OPT})} = \frac{w(S_{2k})}{w(S^{OPT})} \geq 1 - \prod_{i=1}^{2T}(1 - \frac{1}{T}) = 1 - (1 - \frac{1}{T})^{2T} \geq 1 - e^{-2}.$$

Secondly, we use the simple random max-bisection algorithm to find $S^1$ and $S^2$ within $S$, where $|S^1| = |S^2| = T$ and $E(cut(S^1, S^2)) \geq \frac{1}{2}I(S)$ (e.g., see [16]). Then, from Lemma 1,

$$\max\{E(cut(S^1, V \setminus S^1)), E(cut(S^2, V \setminus S^2))\} \geq \frac{1}{2}w(S) \geq \frac{1}{2}(1 - e^{-2})w^*.$$

## References


[1] A. A. Ageev and M. I. Sviridenko, "Approximation Algorithms for Maximum Coverage and Max Cut with Given Sizes of Parts," *IPCO*, G. Cornuejols, R. E. Burkard, G. J. Woeginger (Eds.) pp. 17-30, 1999.

[2] R. Bar-Yehuda, "Using Homogeneous Weights for Approximating the Partial Cover Problem," *10th SODA*, 71-75, 1999.

[3] A. Frieze and M. Jerrum, "Improved Approximation Algorithms for Max $k$-Cut and Max Bisection," *Proc. 4th IPCO Conference*, 1-13, 1995.

[4] E. Halperin and U. Zwick, "Improved Approximation Algorithms for Maximum Graph Bisection Problems," Manuscript, 2000.

[5] Q. Han, Y. Ye, and J. Zhang, "An Improved Rounding Method and Semidefinite Relaxation for Graph Partition," Working Paper, Department of Management Sciences, The University of Iowa, 2000.

[6] R. Hassin and S. Rubinstein, "Approximation Algorithms for Maximum Linear Arrangement," to appear in *Information Processing Letters.*

[7] T.Y. Kao and E.A. Elsayed, "Minimization of the Mean Flow Time of Jobs on Two Parallel Machines with a Machine Availability Constraint," Working Paper 89-104, Department of Industrial Engineering, Rutgers University, New Brunswick, NJ, 1989.

[8] S. Khuller, A. Moss, and J. Naor, "The Budgeted Maximum Coverage Problem," *Information Procession Letters*, 70 (1), 39-45, 1999.

[9] C.-Y. Lee, "Machine Scheduling with an Availability Constraint," *Journal of Global Optimization*, 9: 363-382, 1996.

[10] C.-Y. Lee, L. Lei, M. Pinedo, "Current Trends in Deterministic Scheduling," *Annal of Operations Research*, 70, 1-42, 1997.

[11] C.-Y. Lee and S.D. Liman, "Capacitated Two-Parallel Machines Scheduling to Minimize Sum of Job Completion Times," *Discrete Applied Mathematics*, 41, 211-222, 1993.

[12] S. Sahni, "Algorithms for Scheduling Independent Tasks," *Journal of ACM*, 23, 116-127, 1976.

[13] E. Sanlaville, G. Schmidt, "Machine Scheduling with Availability Constraints," *Acta Informatica*, 35, 795-811, 1998.

[14] M. Skutella, "Semidefinite Relaxations for Parallel Machine Scheduling," in *Proceedings of the 39th Annual IEEE Symposium on Foundations of Computer Science (FOCS'98)*, 472-481, 1998.

[15] H. Yang, Y. Ye and J. Zhang, "An Approximation Algorithm for the Two-Parallel Machines Scheduling Problem with Capacity Constraints", Working Paper, Department of Management Sciences, The University of Iowa, 2000.

[16] Y. Ye, "A .699-approximation Algorithm for Max-Bisection", to appear in *Mathematical Programming.*