Journal of Global Optimization **27:** 399–410, 2003.

Improved Approximation Algorithms for MAX $\frac{n}{2}$-DIRECTED-BISECTION and MAX $\frac{n}{2}$-DENSE-SUBGRAPH

DACHUAN XU[1], JIYE HAN[1], ZHENGHAI HUANG[1] and LIPING ZHANG[2]
[1]*Institute of Applied Mathematics, Academy of Mathematics and System Sciences, Chinese Academy of Sciences, P.O. Box 2734, Beijing 100080, P.R. China (e-mail: xudc@lsec.cc.ac.cn, jiyehan@sina.com, zhhuang@sina.com)*
[2]*Department of Mathematical Sciences, Tsinghua University, Beijing 100084, P.R. China (e-mail: lzhang@math.tsinghua.edu.cn)*

(Received 15 August 2000; accepted in revised form 20 March 2003)

Abstract. We consider the MAX $\frac{n}{2}$-DIRECTED-BISECTION problem, i.e., partitioning the vertices of a directed graph into two blocks of equal cardinality so as to maximize the total weight of the edges in the directed cut. A polynomial approximation algorithm using a semidefinite relaxation with 0.6458 performance guarantee is presented for the problem. The previous best-known results for approximating this problem are 0.5 using a linear programming relaxation, 0.6440 using a semidefinite relaxation. We also consider the MAX $\frac{n}{2}$-DENSE-SUBGRAPH problem, i.e., determine a block of half the number of vertices from a weighted undirected graph such that the sum of the edge weights, within the subgraph induced by the block, is maximized. We present an 0.6236 approximation of the problem as opposed to 0.6221 of Halperin and Zwick.

Key words: MAX $\frac{n}{2}$-DIRECTED-BISECTION, MAX $\frac{n}{2}$-DENSE-SUBGRAPH, Polynomial-time approximation algorithm, Semidefinite programming.

1. Introduction

Given a directed graph $G = (V, A)$, a weight function $w : A \to R^+$ (if $(i, j) \notin A$, then $w_{ij} = 0$), a directed cut in G is defined to be the set of arcs leaving some vertex subset S (we denote it by $\tilde{\delta}(S)$). The maximum directed cut problem (MAX DICUT) is that of finding a directed cut $\tilde{\delta}(S)$ with maximum total weight. In this paper, we consider a version of MAX DICUT (MAX $\frac{n}{2}$-DIRECTED-BISECTION or MDB), and it is required to find a directed cut $\tilde{\delta}(S)$ having maximum weight over all cuts $\tilde{\delta}(S)$ with $|S| = \dfrac{n}{2}$. In the MDB problem, $n = |V|$ is assumed to be even.

MAX DICUT is well-known to be NP-hard and so is MAX $\frac{n}{2}$-DIRECTED-BISECTION. This means that one should not expect to find a polynomial time algorithm for solving it exactly. Therefore many experts are interested in developing polynomial time approximation algorithms for MDB. A (randomized) algorithm

for the maximization problem is called (randomized) r-approximation algorithm, where $0 < r \leq 1$, if it outputs a feasible solution with its (expected) value at least r times the optimum value for all instances of the problem.

Papadimitriou and Yannakakis [15] developed an approximation algorithm for MAX DICUT with an approximation ratio of 0.25. Using a novel technique of rounding semidefinite programming (SDP) relaxations, Goemans and Williamson [8] worked out an algorithm solving MAX DICUT approximately within a factor of 0.796. A bit latter Feige and Goemans [3] developed an algorithm for MAX DICUT with a better approximation ratio of 0.859. Using a method of rounding linear relaxations, Ageev et al. [1] developed an 0.5-approximation algorithm for MAX DICUT with given sizes of parts. Halperin and Zwick [9] improved this result to 0.6440 for MDB.

We also consider the MAX $\frac{n}{2}$-DENSE-SUBGRAPH problem (DSP), i.e., determine a block of half number vertices from a weighted undirected graph such that the sum of the edge weights, within the subgraph induced by the block, is maximized. For DSP, Ye and Zhang [18], using a new SDP relaxation, obtained an improved 0.586 performance guarantee from 0.5 of Feige and Seltser [6] and Goemans [7]. Halperin and Zwick [9] improved this result to 0.6221.

SDP relaxations have been successfully applied to various graph optimization problems [2–11], [16–19]. Combining the previous various techniques, i.e., relaxation of integer programs into semidefinite programs (cf. [12]), mapping (cf. [3]), outward rotations (cf. [14, 17, 19]), random hyperplane rounding (cf. [2, 8]), linear randomized rounding (cf. [9]), and swapping (cf. [16]), we obtain the performance guarantee 0.6458 for MDB and 0.6236 for DSP.

In the following section, we present the algorithm of Halperin and Zwick [9] first, and then introduce the new idea used in this paper. In Sections 3 and 4, we give the performance guarantee for MDB and DSP, respectively. Some discussions are given in Section 5.

2. Halperin-Zwick Algorithm and Our Improved Algorithm

Since our algorithm and analysis closely follows that of Halperin and Zwick [9], we first review the Halperin and Zwick method, and then proceed with our improved method.

By introducing a "reference" binary variable x_0, the MDB problem can be formulated as the following $0-1$ integer program

$$\begin{aligned} w^* := \ \max \ & \frac{1}{4} \sum_{(i,j)\in A} w_{ij}(1 + x_0x_i - x_0x_j - x_ix_j) \\ \text{s. t.} \ & \sum_{j=1}^{n} x_0x_j = 0 \\ & x_j^2 = 1, \ \ j = 0, 1, \cdots, n, \end{aligned} \tag{2.1}$$

where $i \in S$ if and only if $x_i = x_0$ for $i = 1, 2, \cdots, n$.

Using the "triangle inequalities", they introduce a relaxation of (2.1) into a semidefinite program of the following form:

$$\begin{aligned} w^{\mathrm{SDP}} := \ \max \ & \frac{1}{4} \sum_{(i,j)\in A} w_{ij}(1 + v_0 \cdot v_i - v_0 \cdot v_j - v_i \cdot v_j) \\ \text{s. t.} \ & \sum_{1\le i,j\le n} v_i \cdot v_j = 0 \\ & v_i \cdot v_j + v_i \cdot v_k + v_j \cdot v_k \ge -1, \quad 0 \le i, j, k \le n \\ & -v_i \cdot v_j - v_i \cdot v_k + v_j \cdot v_k \ge -1, \quad 0 \le i, j, k \le n \\ & -v_i \cdot v_j + v_i \cdot v_k - v_j \cdot v_k \ge -1, \quad 0 \le i, j, k \le n \\ & v_i \cdot v_j - v_i \cdot v_k - v_j \cdot v_k \ge -1, \quad 0 \le i, j, k \le n \\ & ||v_j|| = 1, \quad v_j \in R^{n+1}, \quad j = 0, 1, \cdots, n. \end{aligned} \tag{2.2}$$

Obviously, (2.2) is a relaxation of (2.1), so that $w^{\mathrm{SDP}} \ge w^*$. Notice that by the constraints of the above program, $||\sum_{i=1}^n v_i||^2 = \sum_{1\le i,j\le n} v_i \cdot v_j = 0$, and therefore $\sum_{i=1}^n v_i = 0$. For $i \ge 1$, the vector v_i corresponds to the vertex i in the graph, and the vector v_0 will be a special vector that will break the symmetry in the MDB.

For any $U \subset V$, denote the total weights within the directed cut $\delta(U)$ as $w(U)$, that is:

$$w(U) := \sum_{(i,j)\in A, i\in U, j\in V\setminus U} w_{ij}.$$

They present a generic approximation algorithm as follows. Here, we revise slightly their algorithm on some parameters (cf. [9] and [11]).

Halperin-Zwick Algorithm

Step 0. **Initialization:** Choose parameters $-1 \le b \le 1, 0 \le c$, a rotation coefficient $0 \le \rho \le 1$, a probability $0 \le \nu \le 1$, and a tolerance $\varepsilon > 0$. Let $\mu = \alpha + c(b\beta_1 + \beta_2)$. (The meaning of these parameters is explained later.)

Step 1. **SDP Solving:** Solve (2.2) and obtain the vectors $v_0, v_1, \cdots, v_n$.

Step 2. **Rotating:** Rotate the vectors $v_0, v_1, \cdots, v_n$ into new unit vectors $w_0', w_1', \cdots, w_n'$ such that $w_i' \cdot w_j' = \rho(v_i \cdot v_j)$.

Step 3. **Randomized Rounding:** Repeat

(a) Choose a random vector r and let $S = \{i \ge 1 : \mathrm{sgn}(w_i' \cdot r) = \mathrm{sgn}(w_0' \cdot r)\}$ and $T = V - S$, where $\mathrm{sgn}(x) = 1$ if $x \ge 0$, and -1 otherwise.

(b) With probability ν override the choice of S made above, and for each $i \in V$, put i in S, independently, with probability $(1 + v_0 \cdot v_i)/2$, and in T, otherwise.

(c) Let $z = \dfrac{w(S)}{w^*} + bc\dfrac{|S|}{n} + c\dfrac{|S||T|}{n^2}$.
Until $z \geq (1-\varepsilon)\mu$.

Step 4. **Vertex Swapping:** Using the random algorithm, correct S to a set $\tilde{S}$ of size $\frac{n}{2}$.

The randomized rounding can be derandomized (cf. [13]). The new algorithmic contributions in this paper are as follows. We use the idea of "mapping" the optimal vectors before performing the "rotation". This idea was previously used by Feige and Goemans [3] for Max 2-SAT and Max directed Cut. Feige and Langberg [4] also used the technique for Max-k-Vertex Cover. We replace Step 2 of Halperin-Zwick algorithm by

Step 2. **Mapping and Rotating:** Map v_i to a vector w_i depending on v_0 and v_i. Let $w_0 := v_0$. Then rotate the vectors $w_0, w_1, \cdots, w_n$ into new unit vectors $w'_0, w'_1, \cdots, w'_n$ such that $w'_i \cdot w'_j = \rho(w_i \cdot w_j)$.

And we denote the improved algorithm as Algorithm 2.1.

Feige and Goemans [3] introduce the mapping of the following form. Map any vector v_i to a vector w_i, coplanar with v_0, on the same side of v_0 as v_i is, and which forms an angle with v_0 equal to $f(\theta_i)$ for some function f, where θ_i is the angle between v_0 and v_i. They impose that $f(\pi - \theta) = \pi - f(\theta)$ to guarantee that the nodes of S are treated in the same manner as the nodes of $V \setminus S$. The original scheme of Goemans and Williamson [8] corresponds to $f_0(\theta) = \theta$. Set $f_1(\theta) = \frac{\pi}{2}(1 - \cos(\theta))$. And letting $f(\theta)$ be a convex combination of $f_0(\theta)$ and $f_1(\theta)$, i.e.,

$$f(\theta) = f(\theta; \eta) := \eta\theta + (1-\eta)\left[\frac{\pi}{2}(1-\cos(\theta))\right], \tag{2.3}$$

where $\eta \in [0, 1]$ is a parameter and will be specified later in the analysis of the algorithm.

In the numerical computations, given θ_i, θ_j and $v_i \cdot v_j$, we need to be able to compute the angle between w_i and w_j in order to compute the probability. This can be done by using the cosine rule for spherical triangles. In particular, we have that

$$\begin{aligned} &w_i \cdot w_0 = v_i \cdot v_0; \\ &v_i \cdot v_j = \cos(\theta_i)\cos(\theta_j) + \cos(\tilde{\theta})\sin(\theta_i)\sin(\theta_j), \\ &w_i \cdot w_j = \cos f(\theta_i)\cos f(\theta_j) + \cos(\tilde{\theta})\sin f(\theta_i)\sin f(\theta_j), \end{aligned} \tag{2.4}$$

where $\tilde{\theta}$ denotes the angle between the planes defined by (v_0, v_i) and (v_0, v_j). This allows the determination of the angle between w_i and w_j. From (2.4), we have

$$w_i \cdot w_j = \cos f(\theta_i)\cos f(\theta_j) + \frac{v_i \cdot v_j - \cos(\theta_i)\cos(\theta_j)}{\sin(\theta_i)\sin(\theta_j)}\sin f(\theta_i)\sin f(\theta_j).$$

3. Approximation of MDB

Since the random algorithm makes the analysis more complicated, we prefer to the following greedy algorithm for MDB in the vertex swapping procedure.

Step 4. **Vertex Swapping:** Let $\tilde{S} = S$. Do one of the following two cases:

If $|\tilde{S}| \geq \frac{n}{2}$, for each $i \in \tilde{S}$, let $\zeta(i) = \sum_{(i,j)\in A,\ j\in V\setminus\tilde{S}} w_{ij}$ and $\tilde{S} := \{i_1, i_2, \cdots, i_{|\tilde{S}|}\}$, where $\zeta(i_1) \geq \zeta(i_2) \geq \cdots \geq \zeta(i_{|\tilde{S}|})$. Then, remove vertex $i_{|\tilde{S}|}$ from $\tilde{S}$ and reassign $\tilde{S} := \{i_1, i_2, \cdots, i_{|\tilde{S}|-1}\}$. Repeat this swapping procedure till $|\tilde{S}| = \frac{n}{2}$.

If $|\tilde{S}| < \frac{n}{2}$, then for each $i \in V \setminus \tilde{S}$, let $\zeta(i) = \sum_{(j,i)\in A,\ j\in\tilde{S}} w_{ji}$ and $V \setminus \tilde{S} = \{i_1, i_2, \cdots, i_{|V\setminus\tilde{S}|}\}$, where $\zeta(i_1) \geq \zeta(i_2) \geq \cdots \geq \zeta(i_{|V\setminus\tilde{S}|})$. Then, add node $i_{|V\setminus\tilde{S}|}$ to $\tilde{S}$ and reassign $V \setminus \tilde{S} := \{i_1, i_2, \cdots, i_{|V\setminus\tilde{S}|-1}\}$. Repeat this swapping process till $|\tilde{S}| = \frac{n}{2}$.

Clearly, the construction of bisection $\tilde{S}$ guarantees that

Lemma 3.1.

$$w(\tilde{S}) \geq \begin{cases} \dfrac{n/2}{|S|} w(S) & \text{if} \quad |S| \geq \dfrac{n}{2}, \\ \dfrac{n/2}{n-|S|} w(S) & \text{if} \quad |S| < \dfrac{n}{2}. \end{cases}$$

Given a vector r drawn uniformly from the unit sphere, we know by the linearity of expectation that (cf. [8])

$$\begin{aligned} E[w(S)|\nu = 0] &= \sum_{(i,j)\in A} w_{ij} \cdot \Pr[\operatorname{sgn}(w_i' \cdot r) = \operatorname{sgn}(w_0' \cdot r) = -\operatorname{sgn}(w_j' \cdot r)] \\ &= \sum_{(i,j)\in A} w_{ij} \cdot \Big[1 - \frac{1}{2\pi}(\arccos(w_i' \cdot w_0') + \arccos(-w_j' \cdot w_0') \\ &\quad + \arccos(-w_i' \cdot w_j'))\Big] \\ &= \sum_{(i,j)\in A} w_{ij} \cdot \Big[- \frac{1}{2\pi}(\arccos(\rho \cos f(\theta_i)) - \arccos(\rho \cos f(\theta_j)) \\ &\quad - \arccos(\rho(w_i \cdot w_j)))\Big]. \end{aligned}$$

For given $\eta, \rho, \nu \in [0, 1]$, let

$$\alpha = \alpha(\eta, \rho, \nu) := \min_{(x,y,z)\in\mathcal{F}} \frac{(1-\nu)h_0(x, y, z) + \nu h_1(x, y, z)}{\frac{1}{4}(1 + \cos x - \cos y - \cos z)}$$

where

$$h_0(x, y, z) = -\frac{1}{2\pi}(\arccos(\rho \cos f(x)) - \arccos(\rho \cos f(y)) - \arccos(\rho g(x, y, z))),$$

$$g(x, y, z) = \cos f(x) \cos f(y) + \frac{\cos z - \cos x \cos y}{\sin x \sin y} \sin f(x) \sin f(y),$$

$$h_1(x, y, z) = \frac{(1 + \cos x)(1 - \cos y)}{4},$$

$$\mathcal{F} := \left\{ (x, y, z) \;\middle|\; \begin{array}{l} \begin{pmatrix} 1 & \cos x & \cos y \\ \cos x & 1 & \cos z \\ \cos y & \cos z & 1 \end{pmatrix} \succeq 0, \\ \cos x + \cos y + \cos z \geq -1, \\ -\cos x - \cos y + \cos z \geq -1, \\ -\cos x + \cos y - \cos z \geq -1, \\ \cos x - \cos y - \cos z \geq -1, \\ 0 \leq x, y, z \leq \pi. \end{array} \right\}.$$

Then, we have the following lemma:

Lemma 3.2. *The expectation of $w(S)$ of Algorithm 2.1 satisfies the following inequality*

$$E[w(S)] \geq \alpha \cdot w^{\text{SDP}} \geq \alpha \cdot w^*.$$

If $\nu = 0$, we have by the linearity of the expectation that

$$\begin{aligned} E[|S|] &= \sum_{i=1}^{n} \Pr\{\text{sgn}(w_i' \cdot r) = \text{sgn}(w_j' \cdot r)\} \\ &= \sum_{i=1}^{n} \left(1 - \frac{\arccos(w_0' \cdot w_i')}{\pi}\right) \\ &= \sum_{i=1}^{n} \left(\frac{1}{2} + \frac{1}{\pi} \arcsin(\rho(w_0 \cdot w_i))\right), \end{aligned} \tag{3.1}$$

$$\begin{aligned} E[|S||T|] &= \sum_{i<j} \Pr\{\text{sgn}(w_i' \cdot r) \neq \text{sgn}(w_j' \cdot r)\} \\ &= \sum_{i<j} \frac{\arccos(w_i' \cdot w_j')}{\pi} \\ &= \sum_{i<j} \frac{\arccos(\rho(w_i \cdot w_j))}{\pi}. \end{aligned} \tag{3.2}$$

If $\nu = 1$, we have (cf. [9])

$$E\left[\frac{|S|}{n}\right] = \frac{1}{n}\sum_{i=1}^{n} \Pr\{i \in S\} = \frac{1}{2}, \tag{3.3}$$

$$E\left[\frac{|S||T|}{n^2}\right] = \frac{1}{n^2}\sum_{i,j} \Pr\{i \in S,\, j \in T\} \geq \frac{1}{4} - \frac{1}{2n}. \tag{3.4}$$

For given $\eta, \rho \in [0, 1]$, let

$$\tau_1 = \min_{0 \leq x \leq \pi} \frac{\arcsin(\rho) - \arcsin(\rho \cos f(x))}{1 - \cos x}, \tag{3.5}$$

$$\beta = \frac{2}{\pi}[\arccos(\rho) + \tau_1], \tag{3.6}$$

$$\tau_2 = \tau_2(\eta, \rho) := \min_{(x,y,z) \in \mathcal{F}} \frac{\arccos(\rho g(x, y, z)) - \arccos(\rho)}{1 - \cos z}, \tag{3.7}$$

$$\gamma = \frac{2}{\pi}\left[\arccos(\rho) + \tau_2 - \frac{\arccos(\rho)}{n}\right]. \tag{3.8}$$

From (3.1)–(3.8) and Lemma 1 of [11], we have the following lemma:

Lemma 3.3. *Let*

$$\beta_1 = \begin{cases} \frac{(1-\nu)\beta+\nu}{2} & \text{if } b \geq 0, \\ \frac{(1-\nu)(2-\beta)+\nu}{2} & \text{otherwise;} \end{cases}$$

and

$$\beta_2 = (1 - \nu) \cdot \frac{\gamma}{4} + \nu \cdot \left(\frac{1}{4} - \frac{1}{2n}\right).$$

Then, Algorithm 2.1 yields S satisfying the following inequalities:

$$E\left[b\frac{|S|}{n}\right] \geq b\beta_1,$$

$$E\left[\frac{|S||T|}{n^2}\right] \geq \beta_2.$$

Define a new random variable as

$$z(b, c) := \frac{w(S)}{w^*} + bc\frac{|S|}{n} + c\frac{|S||T|}{n^2}, \quad -1 \le b \le 1, c \ge 0. \tag{3.9}$$

For MDB, we set $b = 0$ which implies that we needn't compute β_1 in this section. Then we have

$$E[z(c)] \ge \alpha + c\beta_2.$$

Lemma 3.4. *If random variable $z(c)$ fulfills its expectation, i.e., $z(c) \ge \alpha + c\beta_2$, then*

$$w(\tilde{S}) \ge R(\eta, \rho, \nu, c) \cdot w^*, \tag{3.10}$$

where

$$R(\eta, \rho, \nu, c) = \begin{cases} 2(\sqrt{c(\alpha + c\beta_2)} - c) & \text{if } 1 \le \sqrt{\dfrac{\alpha + c\beta_2}{c}} \le 2, \\ \min\{\alpha + c\beta_2 - c, \dfrac{1}{2}(\alpha + c\beta_2)\} & \text{otherwise.} \end{cases}$$

Proof. Suppose that

$$w(S) = \lambda w^* \quad \text{and} \quad |S| = \delta n.$$

Note that $\delta \in [0, 1]$. It follows from $z(c) \ge \alpha + c\beta_2$ that

$$\lambda \ge \alpha + c\beta_2 - 4c\delta(1 - \delta). \tag{3.11}$$

Now we consider two possibilities, $\delta \ge \frac{1}{2}$ and $\delta \le \frac{1}{2}$. From Lemma 3.1, (3.11), and simple calculus, we obtain (3.10) (cf. [18]). □

Finally, we solve the maximization problem

$$\begin{array}{lll} R = & \max & R(\eta, \rho, \nu, c) \\ & \text{s.t.} & 0 \le \eta, \rho, \nu \le 1 \\ & & c \ge 0. \end{array} \tag{3.12}$$

We have played with *Matlab* 6.0 and find $R = 0.6458$ when $\eta = 0.64$, $\rho = 0.96$, $\nu = 0.06$, and $c = 4.6456$. At this case, $\alpha = 0.8247$, $\gamma = 0.9183$, and $\beta_2 = 0.2308$. Together with Lemma 3.4 and Frieze and Jerrum [5]'s analysis, we have the final result:

Theorem 3.5. *The worst-case performance ratio of Algorithm 2.1 for the MDB problem is at least* 0.6458 *for sufficiently large n.*

4. Approximation of DSP

By introducing a "reference" binary variable x_0, the DSP problem can be formulated as the following $0-1$ integer program

$$\begin{aligned} w^* := \ \max \ & \frac{1}{4}\sum_{(i,j)\in A} w_{ij}(1 + x_0x_i + x_0x_j + x_ix_j) \\ \text{s. t.} \ & \sum_{j=1}^{n} x_0x_j = 0 \\ & x_j^2 = 1, \ \ j = 0, 1, \cdots, n, \end{aligned} \tag{4.1}$$

where $i \in S$ if and only if $x_i = x_0$ for $i = 1, 2, \cdots, n$.

Same as (2.2), we can get the SDP relaxation of (4.1). Then apply Algorithm 2.1 for the SDP relaxation of (4.1).

The detailed greedy algorithm for DSP is as follows.

Step 4. **Vertex Swapping:** Let $\tilde{S} = S$. Do one of the following two cases:

If $|\tilde{S}| \geq \frac{n}{2}$, for each $i \in \tilde{S}$, let $\zeta(i) = \sum_{j\in\tilde{S}} w_{ij}$ and $\tilde{S} := \{i_1, i_2, \cdots, i_{|\tilde{S}|}\}$, where $\zeta(i_1) \geq \zeta(i_2) \geq \cdots \geq \zeta(i_{|\tilde{S}|})$. Then, remove vertex $i_{|\tilde{S}|}$ from $\tilde{S}$ and reassign $\tilde{S} := \{i_1, i_2, \cdots, i_{|\tilde{S}|-1}\}$. Repeat this swapping procedure till $|\tilde{S}| = \frac{n}{2}$.

If $|\tilde{S}| < \frac{n}{2}$, arbitrarily add $\frac{n}{2} - |\tilde{S}|$ vertices from outside of $\tilde{S}$ into $\tilde{S}$.

For any $U \subset V$, we redefine $w(U)$ in this section as follows. Denote the total weights within the subgraph induced by U as $w(U)$, that is, $w(U) := \sum_{i<j, i,j\in U} w_{ij}$. Clearly, the construction of bisection $\tilde{S}$ guarantees that (cf. [11])

Lemma 4.1.

$$w(\tilde{S}) \geq \begin{cases} \dfrac{\frac{n}{2}\left(\frac{n}{2}-1\right)}{|S|(|S|-1)} w(S) & \text{if} \quad |S| \geq \dfrac{n}{2}, \\ w(S) & \text{if} \quad |S| < \dfrac{n}{2}. \end{cases}$$

For given $\eta, \rho, \nu \in [0, 1]$, let

$$\alpha = \alpha(\eta, \rho, \nu) := \min_{(x,y,z)\in\mathcal{F}} \frac{(1-\nu)h_0(x, y, z) + \nu h_1(x, y, z)}{\frac{1}{4}(1 + \cos x + \cos y + \cos z)}$$

where

$$\begin{aligned} h_0(x, y, z) =& 1 - \frac{1}{2\pi}(\arccos(\rho\cos f(x)) + \arccos(\rho\cos f(y)) \\ & + \arccos(\rho g(x, y, z))), \end{aligned}$$

$$h_1(x, y, z) = \frac{(1+\cos x)(1+\cos y)}{4}.$$

Then we have Lemma 3.1. Lemma 3.2 also holds for DSP. Restrict $b \in [-1, 0]$ and let

$$c = \begin{cases} \dfrac{\alpha}{1/4 - (b\beta_1 + \beta_2)} & \text{if } b \in [-\frac{1}{2}, 0], \\ \dfrac{3\alpha}{(b\beta_1 + \beta_2) - b + (1+b)^2 - 4(b\beta_1 + \beta_2)} & \text{if } b \in [-1, -\frac{1}{2}). \end{cases}$$

Han et. al. [11] showed the following lemma,

Lemma 4.2. *If random variable $z(b, c)$ fulfills its expectation, i.e., $z(b, c) \geq \alpha + c(b\beta_1 + \beta_2)$, then*

$$w(\tilde{S}) \geq R(\eta, \rho, \nu, b) \cdot w^*, \tag{4.2}$$

where

$$R(\eta, \rho, \nu, b) = \begin{cases} \dfrac{\alpha(1-(1+b)^2)}{1 - 4(b\beta_1 + \beta_2)} & \text{if } b \in [-\frac{1}{2}, 0], \\ \dfrac{\alpha((1+b)^2/4 - b)}{(b\beta_1 + \beta_2) - b + (1+b)^2 - 4(b\beta_1 + \beta_2)} & \text{if } b \in [-1, -\frac{1}{2}]. \end{cases}$$

Finally, we solve the maximization problem

$$\begin{aligned} R = \ & \max \ R(\eta, \rho, \nu, b) \\ & \text{s.t.} \ \ 0 \leq \eta, \rho, \nu \leq 1 \\ & \quad\ \ -1 \leq b \leq 0. \end{aligned} \tag{4.3}$$

We have played with *Matlab* 6.0 and find $R = 0.6236$ when $\eta = 0.77$, $\rho = 0.88$, $\nu = 0.16$, and $b = -0.1797$. At this case, $\alpha = 0.7687$, $\gamma = 0.9526$, $\beta_1 = 0.5056$, $\beta_2 = 0.2400$, and $c = 7.6234$. Together with Lemma 4.2 and Frieze and Jerrum [5]'s analysis, we have the final result:

Theorem 4.3. *The worst-case performance ratio of Algorithm 2.1 for the DSP problem is at least* 0.6236 *for sufficiently large n.*

5. Discussions

Obviously, all the asymmetric graph bisection problems, such as MAX $\frac{n}{2}$-VERTEX-COVER (MVC) and MAX $\frac{n}{2}$-DIRECTED-UNCUT (DUC), may be considered under the frame of Algorithm 2.1. Unfortunately, we find that an improved algorithm for MVC or DUC could not obtained using the current techniques. The reason is that mapping doesn't benefit the balance between α and γ. If we adopt

mapping, the decrease of γ counteracts the advantage of increase of α. That is to say, the value of α is large enough and the most important thing is to increase the value of γ.

The result of 0.6458 for MDB or 0.6236 for DSP is based on the bound of α and τ_2 for given η, ρ, and ν. However, α and τ_2 are defined as the optimal values of some non-convex optimization problems, and are computed by non-rigorous numerical computations (cf. [3]). We utilize repeatedly the "fmincon" function of Optimization Toolbox of $Matlab$ 6.0 to calculate α and τ_2 starting with more initial points which are situated in the cube $[0, \pi] \times [0, \pi] \times [0, \pi]$.

It is an interesting question how to adopt the idea of Halperin and Zwick (2000) about using triangle inequalities to improve the bound of γ. But it seems too complicate to analyze it.

Acknowledgments

The first author would like to thank Yinyu Ye for his useful comments and Qiaoming Han and Jiawei Zhang for their helpful discussions on this paper. We also thank Alexander Ageev, Uriel Feige, and Uri Zwick for kindly giving us their papers. Two referees are also acknowledged for their valuable comments.

This work was partly supported by Chinese NSF grant 10271002 and National 973 Information Technology and High-Performance Software Program of China with grant No. G1998030401. The first author gratefully acknowledges the support of K. C. Wong Education Foundation, Hong Kong.

References

1. Ageev, A., Hassin, R., and Sviridenko M. (2001), An 0.5-approximation algorithm for the max dicut with given sizes of parts, *SIAM J. Discrete Math.* 14, 246–255.
2. Bertsimas, D. and Ye, Y. (1998), Semidefinite relaxations, multivariate normal distributions, and order statistics. In: Du, D. Z. and Pardalos, P.M. (eds.), *Handbook of Combinatorial Optimization* (Vol. 3), pp. 1–19, Kluwer Academic Publishers.
3. Feige, U. and Goemans, M. X. (1995), Approximating the value of two prover proof systems, with applications to MAX 2SAT and MAX DICUT. In: *Proceedings of the 3rd Israel Symposium on Theory and Computing Systems*, Tel Aviv, Israel, pp. 182–189.
4. Feige, U. and Langberg, M. (1999), Approximation algorithms for maximization problems arising in graph partitioning, Manuscript and M. Sc. thesis.
5. Frieze, A. and Jerrum, M. (1997), Improved approximation algorithms for max $k-$cut and max bisection, *Algorithmica* 18, 67-81.
6. Feige, U. and Seltser, M. (1997), On the densest $k-$subgraph problem, Technical Report, Department of Applied Mathematics and Computer Science, The Weizmann Institute, Rehovot.
7. Goemans, M. X. (1996), Mathematical programming and approximation algorithms, Lecture given at the Summer School on Approximate Solution of Hard Combinatorial Problems, Udine.

8. Goemans, M. X. and Williamson, D. P. (1995), Improved approximation algorithms for Maximum Cut and Satisfiability problems using semidefinite programming, *Journal of ACM* 42, 1115–1145.
9. Halperin, E. and Zwick, U. (2001), A unified framework for obtaining improved approximation algorithms for maximum graph bisection problems, Manuscript.
10. Han, Q., Ye, Y., Zhang, H., and Zhang, J. (2002), On approximation of Max-Vertex-Cover, to appear in *European Journal of Operational Research*.
11. Han, Q., Ye, Y., and Zhang, J. (2002), An improved rounding method and semidefinite programming relaxation for graph partition, *Math. Program.* 92, 509–535.
12. Lovasz, L. and Shrijver, A. (1990), Cones of matrices and setfunctions, and 0-1 optimization, *SIAM J. of Optimization* 1, 166–190.
13. Mahajan, S. and Ramesh, H. (1999), Derandomizing approximation algorithms based on semidefinite programming, *SIAM J. Comput.* 28, 1641–1663.
14. Nesterov, Y. (1998), Semidefinite relaxation and nonconvex quadratic optimization, *Optimization Methods and Software* 9, 141–160.
15. Papadimitriou, C. H. and Yannakakis, M. (1991), Optimization, approximation, and complexity classes, *J. Comput. Syst. Sci.* 43, 425–440.
16. Srivastav, A. and Wolf, K. (1998), Finding dense subgraphs with semidefinite programming. In: Jansen, K. and Rolim, J. (eds.) *Approximation Algorithms for Combinatorial Optimization*, pp. 181–191.
17. Ye, Y. (2001), A .699-approximation algorithm for Max-Bisection, *Math. Program.* 90, 101–111.
18. Ye, Y. and Zhang, J. (2003), Approximation of dense-$\frac{n}{2}$-subgraph and the complement of min-bisection, *Journal of Global Optimization* 25, 55–73.
19. Zwick, U. (1999), Outward rotations: a tool for rounding solutions of semidefinite programming relaxations, with applications to max cut and other problems. In: *Proceedings of the 30th Symposium on Theory of Computation (STOC)*, pp. 551–560.