Journal of Computational Mathematics, Vol.21, No.3, 2003, 357–366.

APPROXIMATION ALGORITHM FOR MAX-BISECTION PROBLEM WITH THE POSITIVE SEMIDEFINITE RELAXATION[*1)]

Da-chuan Xu Ji-ye Han
(*Institute of Applied Mathematics, Academy of Mathematics and System Sciences, Chinese Academy of Sciences, Beijing 100080, China*)

Abstract

Using outward rotations, we obtain an approximation algorithm for Max-Bisection problem, i.e., partitioning the vertices of an undirected graph into two blocks of equal cardinality so as to maximize the weights of crossing edges. In many interesting cases, the algorithm performs better than the algorithms of Ye and of Halperin and Zwick. The main tool used to obtain this result is semidefinite programming.

Key words: Approximation algorithm, Max-Bisection problem, Semidefinite programming, Approximation ratio.

1. Introduction

Given an undirected graph $G = (V, E)$ and nonnegative weights $w_{ij} = w_{ji}$ on the edges $(i, j) \in E$, the maximum cut problem (Max-Cut) is that of finding the set of vertices S that maximizes the weight of the edges in the *cut* $(S, V \setminus S)$; that is, the weight of the edges with one endpoint in S and the other in $V \setminus S$. For simplicity, we usually set $w_{ij} = 0$ for $(i, j) \notin E$ and denote the weight of a cut $(S, \bar{S})$ by $w(S) = \sum_{i \in S, j \in V \setminus S} w_{ij}$. The Maximum Bisection of $G = (V, E)$ is a partition of the vertex set V into two equally sized sets S and $V \setminus S$ so that $w(S)$ is maximized. In Max-Bisection (MB) problem, $n = |V|$ is assumed to be even.

Max-Cut is well-known to be NP-hard and so is Max-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 Max-Cut and Max-Bisection. 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.

The best known approximation ratio for Max-Cut is 0.87856 ([7]). Using outward rotations, Zwick [14] obtain an approximation algorithm for Max-Cut that, in many interesting cases, the algorithm performs better than the algorithm of Goemans and Williamson [7]. Feige, Karpinski and Langberg [5] show that for graphs of degree at most Δ, their algorithm achieces an approximation ratio of at least $0.87856 + \varepsilon$, where $\varepsilon > 0$ is a constant that depends only on Δ. In particular, using computer assisted analysis, they show that for graphs of maximal degree 3, their algorithm obtains an approximation ratio of at least 0.921, and for 3-regular graphs, the approximation ratio is at least 0.924. We refer to [2] for the survey of Max-Cut.

[*] Received January 6, 2001; final revised October 11, 2001.

[1)] Research partly supported by Chinese NSF grant 19731001 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.

To our knowledge, the best known approximation ratio for Max-Bisection is 0.7016 ([8]). This work is an extension of the works of [1] and [12]. Feige Karpinski and Langberg [4] design a 0.795 approximation algorithm for Max-Bisection restricted to regular graphs. In the case of three regular graphs their results imply an approximation ratio of 0.834.

All the above works adopt the SDP relaxation for Max-Cut or Max-Bisection. SDP relaxations have been successfully applied to various graph optimization problems ([1]-[10], [12]-[14]). Many experts put forward different techiques to get a good approximation solution from the SDP relaxation (for example, see [7], [8], [12], and [14]). Using these techiques carefully, we obtain an approximation algorithm for Max-Bisection that, in many interesting cases, the algorithm performs better than the algorithm of Halperin and Zwick [8].

2. SDP Relaxation of MB

The MB problem can be formulated as the following binary integer program

$$\begin{aligned} w^* := \quad & \text{Max} \quad \frac{1}{2}\sum_{i<j} w_{ij}(1 - x_i x_j) \\ & \text{s.t.} \quad \sum_{j=1}^{n} x_j = 0 \\ & \qquad x_j^2 = 1, \ j = 1, \cdots, n. \end{aligned} \tag{2.1}$$

Using the "triangle inequalities", the SDP relaxation of MB becomes

$$\begin{aligned} w^{\text{SDP}} := \quad & \text{Max} \quad \frac{1}{2}\sum_{i<j} w_{ij}(1 - v_i \cdot v_j) \\ & \text{s.t.} \quad \sum_{1 \le i,j \le n} v_i \cdot v_j = 0, \\ & \qquad v_i \cdot v_j + v_i \cdot v_k + v_j \cdot v_k \ge -1, \quad 1 \le i,j,k \le n \\ & \qquad -v_i \cdot v_j - v_i \cdot v_k + v_j \cdot v_k \ge -1, \quad 1 \le i,j,k \le n \\ & \qquad -v_i \cdot v_j + v_i \cdot v_k - v_j \cdot v_k \ge -1, \quad 1 \le i,j,k \le n \\ & \qquad v_i \cdot v_j - v_i \cdot v_k - v_j \cdot v_k \ge -1, \quad 1 \le i,j,k \le n \\ & \qquad ||v_j|| = 1, \quad v_j \in R^{n+1}, \quad j = 1, \cdots, n. \end{aligned} \tag{2.2}$$

To state it precisely, it is convenient to use the following notation.

$W = \sum_{i<j} w_{ij}$ — The total weight of the edges of the graph.

$A = w^{\text{SDP}}/W$ — The ratio between the solution of the relaxation (2.2) and the total weight of the edges in the graph.

$B = w^*/W$ — The ratio between the solution of the relaxation (2.1) and the total weight of the edges in the graph.

Clearly $w^* \le w^{\text{SDP}} \le W$. By inductive method, one can prove that $w^* \ge W/2$. So we get that $1/2 \le B \le A \le 1$.

We present an SDP-based approximation algorithm for MB as follows.

SDP-Algorithm

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

Step 2. **Rotating:** Rotate the vectors $v_1, \cdots, v_n$ into new vectors $v_1', \cdots, v_n'$.

Step 3. **Randomized Rounding:** Choose a random hyperplane (by choosing randomly its normal r which is a vector uniformly distributed on the unit sphere), and set $S = \{i | v_i' \cdot r \ge 0\}$.