Improved Approximation Algorithms for the Partial Vertex Cover Problem

Eran Halperin[1]* and Aravind Srinivasan[2]**

[1] CS Division, Soda Hall, University of California Berkeley, CA 94720-1776
eran@eecs.berkeley.edu
[2] Department of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742 srin@cs.umd.edu

Abstract. The partial vertex cover problem is a generalization of the vertex cover problem: given an undirected graph $G = (V, E)$ and an integer k, we wish to choose a minimum number of vertices such that at least k edges are covered. Just as for vertex cover, 2-approximation algorithms are known for this problem, and it is of interest to see if we can do better than this. The current-best approximation ratio for partial vertex cover, when parameterized by the maximum degree d of G, is $(2 - \Theta(1/d))$. We improve on this by presenting a $(2 - \Theta(\frac{\ln \ln d}{\ln d}))$-approximation algorithm for partial vertex cover using semidefinite programming, matching the current-best bound for vertex cover. Our algorithm uses a new rounding technique, which involves a delicate probabilistic analysis.

Key words and phrases. Partial vertex cover, approximation algorithms, semidefinite programming, randomized rounding

1 Introduction

The vertex cover problem is basic in combinatorial optimization, and is as follows. Given an undirected graph $G = (V, E)$, we wish to choose a minimum number of vertices such that every edge is *covered* (has at least one end-point chosen). In recent years, *partial covering* versions of such covering problems have received quite a bit of attention. For instance, a partial vertex cover problem gets as input a graph G and an integer k, and we aim to choose a minimum number of vertices such that at least k edges are covered. Just as for the vertex cover problem, 2-approximation algorithms are known for this problem [2, 7, 1], and it is an interesting question if we can do better than a factor of 2. The current-best approximation ratio for partial vertex cover, when parameterized by the maximum degree d of G, is $(2 - \Theta(1/d))$ [4, 10]; such a bound is unavoidable if we use the (natural) linear programming (LP) relaxation considered in [4, 10]. In this paper, we develop a $(2 - \Theta(\frac{\ln \ln d}{\ln d}))$-approximation algorithm for partial vertex

* Supported in part by NSF grants CCR-9820951 and CCR-0121555 and DARPA cooperative agreement F30602-00-2-0601.
** Supported in part by NSF Award CCR-0208005.

cover via semidefinite programming (SDP), matching the current-best bound for the standard "full coverage" version of vertex cover (wherein k equals $|E|$) [5].

While the work of [5] on vertex cover also uses SDP, our work differs from it in some crucial ways. At a high level, just as in [5], the SDP outputs a unit vector v_0 and, for each vertex i, a unit vector v_i; ideally, each v_i (for $i \in V$) would be of the form $\pm v_0$, naturally indicating whether to place i in the vertex cover or not, but the SDP can of course output an arbitrary collection of unit vectors v_i. Nevertheless, as pointed out in [5], the condition that every edge must be covered in the vertex cover problem leads to some good valid constraints. For instance, the troublesome case for the vertex cover problem is when, for some edge (i, j), the vectors v_i and v_j are roughly orthogonal to v_0. However, since every edge must be covered in vertex cover, this forces v_i to be approximately $-v_j$, a fact which can then be exploited [5]. However, in our partial cover setting, an edge may only be partially covered; this leads to at least two different troublesome cases, which need to be handled simultaneously. Also, as opposed to current algorithms based on SDP, our rounding scheme does a "semidefinite rounding" and also interprets some of the variables as components of an LP and does an "LP-style" rounding on them. Finally, our algorithm succeeds with a small probability such as $\Theta(1/\log|V|)$ and hence, the analysis needs to be somewhat delicate; in particular, we need certain types of random variables to be very well concentrated around their means, for which we use recently-developed ideas from [10]. (The success probability of $\Omega(1/\log|V|)$ can then be boosted by repetition.)

The rest of this paper is organized as follows. We present the known LP relaxation and our SDP relaxation in Section 2. Some useful tools are then described in Section 3. Our rounding algorithm and its analysis are presented in Sections 4 and 5 respectively. Section 6 then shows some extensions of our main result, and concluding remarks are made in Section 7.

2 Relaxations of the Partial Vertex Cover Problem

Let $G = (V, E)$ be a graph with $|E| = m$ edges and $|V| = n$ vertices. Let $S \subseteq V$ be a subset of the vertices of G. We say that S covers an edge $(u, v) \in E$ if $u \in S$ or $v \in S$. We say that S is a k-cover of G if S covers at least k edges. In the partial vertex cover problem, we are given a graph $G = (V, E)$ with $|E| = m$ and $|V| = n$, and a number $k \leq m$, and are interested in finding a minimum size k-cover of G.

Let $V = \{1, \ldots, n\}$, and consider the following LP relaxation of the partial vertex cover problem.

$$\boxed{\begin{array}{ll} \text{Minimize} \displaystyle\sum_{i=1}^{n} x_i & \\ \quad 0 \leq x_i \leq 1 \quad , & i \in V \\ \quad 0 \leq z_{ij} \leq 1 \quad , & (i,j) \in E \\ \quad z_{ij} \leq x_i + x_j \ , & (i,j) \in E \\ \quad \displaystyle\sum_{(i,j)\in E} z_{ij} \geq k & \end{array}} \tag{1}$$

Relaxation (1) was used in [4] and in [10] to show a 2-approximation to the problem. In fact, these two papers show that using this LP relaxation, one can get a $2-\Theta(1/d)$-approximation, where d is the maximal degree in G. In [10], the rounding procedure takes linear time. The integrality ratio of relaxation (1) is $2 - \frac{1}{n}$: it is attained, e.g., for the complete graph with n vertices, and by setting $k = |E|$. In the complete graph, the maximal degree is $d = n - 1$; therefore, in order to get an approximation better than $(2-\Theta(1/d))$ for the problem, we need a different relaxation.

In [5] it is shown that using semidefinite programming one can achieve a $2 - \frac{(2-o(1))\ln\ln d}{\ln d}$-approximation algorithm for the vertex cover problem; here, the "$o(1)$" term is a function of only d, and tends to 0 as d increases. We extend these ideas by using the following semidefinite relaxation for the partial vertex cover problem. In the following, $\|v\|$ denotes the 2-norm of a vector v.

$$\boxed{\begin{array}{ll} \text{Minimize} \displaystyle\sum_{i=1}^{n} \frac{1 + v_0 \cdot v_i}{2} & \\ \quad \displaystyle\sum_{(i,j)\in E} \frac{3 + v_0 \cdot v_j + v_0 \cdot v_i - v_i \cdot v_j}{4} \geq k & \\ \quad \dfrac{3 + v_0 \cdot v_j + v_0 \cdot v_i - v_i \cdot v_j}{4} \leq 1 \quad , & (i,j) \in E \\ \quad v_0 \cdot v_i + v_0 \cdot v_j + v_i \cdot v_j \geq -1 \quad , & (i,j) \in E \\ \quad v_i \in \mathbb{R}^n, \|v_i\| = 1 \quad , & i \in (\{0\} \cup V) \end{array}} \tag{2}$$

We first show that this is indeed a relaxation. Let $S \subseteq V$ be a k-cover of G. Assign for each vertex $i \in V$ a unit vector v_i. If $i \in S$ we set $v_i = v_0$, and otherwise we set $v_i = -v_0$. It is easy to verify that in this case, for every covered edge $(i,j) \in E$ we have that $\frac{3+v_0 \cdot v_j + v_0 \cdot v_i - v_i \cdot v_j}{4} = 1$, and since S is a k-cover, all the constraints are satisfied. This relaxation can be solved (to within an exponentially small additive error) in polynomial time.

Note that relaxation (2) is an extension of relaxation (1), as one can assign

$$x_i = \frac{1 + v_0 \cdot v_i}{2} \text{ and } z_{ij} = \frac{3 + v_0 \cdot v_j + v_0 \cdot v_i - v_i \cdot v_j}{4}. \tag{3}$$

Thus, every solution of (2) corresponds to a solution of (1), while the other direction is not necessarily true. This implies that the semidefinite constraints are tighter then the linear constraints, and thus, there is hope to gain more by using the semidefinite programming relaxation.

3 Some Useful Ingredients

Before introducing the algorithm, we state some useful lemmas and notations. We will use some basic n-dimensional geometry, and some probability distributions presented in [10].

3.1 Properties of n-dimensional Geometry

Recall that $\phi(x) = \frac{1}{\sqrt{2\pi}} e^{\frac{-x^2}{2}}$ is the density function of a standard normal random variable. Let $N(x)$ be the tail of the standard normal distribution, i.e., $N(x) = \int_x^\infty \phi(y)dy$. We will use the following bounds on $N(x)$ (see Lemma VII.2 of Feller [3]):

Lemma 1. *For every $c \geq 1$, $\phi(c)(\frac{1}{c} - \frac{1}{c^3}) < N(c) < \frac{\phi(c)}{c}$.*

A random vector $r = (r_1, \ldots, r_n)$ is said to have the *n-dimensional standard normal distribution* if the components r_i are independent random variables, each one having the standard normal distribution. In this paper a random n-dimensional vector will always denote a vector chosen from the n-dimensional standard normal distribution. We will use the following (see Theorem IV.16.3 in [9]):

Lemma 2. *Let u be any unit vector in $\mathbb{R}^n$. Let r be a random n-dimensional vector. The projection of r on u, given by the product $r \cdot u$, is distributed according to the standard normal distribution.*

The following lemma is proved in [8] (see also [6]). For completeness, we will give the proof here.

Lemma 3. *Given two unit vectors $u, v \in \mathbb{R}^n$, and a random n-dimensional vector r,*

$$\Pr(u \geq r, v \geq r) \leq N(\frac{2r}{\|u+v\|}).$$

Proof. By Lemma 2,

$$\Pr(u \geq r, v \geq r) \leq \Pr((u+v) \geq 2r) = \Pr(\frac{u+v}{\|u+v\|} \geq \frac{2}{\|u+v\|} r) = N(\frac{2r}{\|u+v\|}).$$

3.2 Certain Distributions on Level-Sets

One of the ingredients of our algorithm is the distribution on level-sets presented in [10]. Suppose we are given a sequence of t numbers $P = (p_1, p_2, \ldots, p_t)$, such that $0 \le p_i \le 1$, and $p_1 + p_2 + \ldots p_t = \ell$ for some integer ℓ. Then, the distribution $\mathcal{D}(t, P)$ is defined to be any distribution on $\{0,1\}^t$, such that if $(X_1, \ldots, X_t) \in \{0,1\}^t$ denotes a vector sampled from $\mathcal{D}(t, P)$, then the following properties hold:

(A1) $\Pr(X_i = 1) = p_i$ for every $i \le t$.
(A2) $\Pr(\sum_{i=1}^t X_i = \ell) = 1$.
(A3) For all $S \subseteq \{1, \ldots, t\}$,

$$\Pr\left(\bigwedge_{i \in S} (X_i = 0)\right) \le \prod_{i \in S} \Pr(X_i = 0)\,; \quad \Pr\left(\bigwedge_{i \in S} (X_i = 1)\right) \le \prod_{i \in S} \Pr(X_i = 1)\,.$$

It is shown in [10] that $\mathcal{D}(t, P)$ exists for any P whose entries add up to an integer, and that we can generate a random sample from $\mathcal{D}(t, P)$ in $O(t)$ time.

In our applications, the condition that $p_1 + p_2 + \ldots p_t$ is an integer may not always hold. To take care of this, we define a distribution $\mathcal{D}'(t, P)$ as follows. Let $\psi \in [0, 1)$ be the smallest non-negative value such that $p_1 + p_2 + \ldots p_t + \psi$ is an integer; define $P' = (p_1, p_2, \ldots, p_t, \psi)$. Then, the distribution $\mathcal{D}'(t, P)$ is defined to be the projection of $\mathcal{D}(t+1, P')$ on to its first t components: i.e., to sample from $\mathcal{D}'(t, P)$, we sample from $\mathcal{D}(t+1, P')$ and discard the last component of the sample. It is easy to check that (A1) and (A3) still hold for the sample from $\mathcal{D}'(t, P)$, and that instead of (A2), we get

(A2') $\Pr((\sum_{i=1}^t X_i) \in \{\lfloor \sum_{i=1}^t p_i \rfloor, \lceil \sum_{i=1}^t p_i \rceil\}) = 1$.

4 A $(2 - \Theta(\frac{\ln \ln d}{\ln d}))$-Approximation Algorithm

We first give an outline of the algorithms given in [4, 10]. Their algorithm is based on LP relaxation (1). We first solve the LP, and then, for each vertex i such that $x_i \ge \frac{1}{2-\epsilon}$, we round x_i to 1; ϵ is a term of the form $\Theta(1/d)$. For each vertex such that $x_i \le \frac{1}{2-\epsilon}$, we round it to 1 with probability $x_i(2-\epsilon)$. The latter is done independently for each vertex in [4] and using a certain distribution on level-sets in [10]. If at least k edges are covered then we are done. Otherwise, we add vertices arbitrarily, each vertex covering at least one new edge, until the number of covered edges is at least k. The analysis needs to show that not many vertices are added in the last step. It turns out that the worst case for such an analysis is when $x_i = x_j = \frac{1}{2}$ for all $(i, j) \in E$. In this case, the LP "pays" one unit for covering the edge (i, j) since $x_i + x_j = 1$, but there is still a small probability that the algorithm will not cover this edge.

Supplied with the ingredients described in the last section, we are now ready to describe our algorithm. The flow of the algorithm is given in Figure 1; please do not confuse the parameter x (which will be $\Theta(\frac{\ln \ln d}{\ln d})$) with the variables x_i

defined in (3). Also, the distributions $\mathcal{D}'(\cdot,\cdot)$ are those defined in Section 3.2. The algorithm starts by solving the semidefinite relaxation. It then partitions the set of vertices into three sets: S_1, S_2, and S_3. S_1 is the set of vectors that are close to v_0, S_3 is the set of vectors that are close to $-v_0$, and S_2 is the set of vectors that are almost orthogonal to v_0. We then take care of each set separately.

We now give some intuition. We first add to the cover all the vertices in S_1. Clearly, since all vertices of S_1 contribute at least $\frac{1+x}{2}$ to the SDP, we get an approximation ratio of $2/(1+x) \approx (2-2x)$ on these. For S_2, consider the following two extreme cases. The first case is an edge (i,j) in $S_2 \times S_2$, such that $z_{ij} = 1$. In this case, since v_i and v_j are almost orthogonal to v_0, it can be shown that $v_i \cdot v_j \approx -1$. Therefore, in step 4, it is not likely that both v_i and v_j will be removed from the cover, and therefore, the edge will be covered with good probability; thus, we can handle the case that was difficult for the LP relaxation. The other extreme case is when $v_i = v_j$, and then $z_{ij} \approx 0.5$. If v_i and v_j are not removed from the cover in step 4 we are fine. Otherwise, by putting each of them back independently with probability $1-\epsilon$, we get that the probability of covering the edge is much more than z_{ij}. Step 5 is taking care of these edges; we use a distribution on level-sets since we need to make sure that the total number of vertices added to the cover is small. In S_3 we take each vertex to be in the cover with probability $\sim (2-4x)x_i$, and since we use a level-sets distribution, we get that the total number of vertices taken there is $\sim (2-4x)$ times the contribution of S_3 to the semidefinite program. Moreover, property (A3) of level-sets distributions can be used to show that the number of edges covered is sufficient with a reasonable probability.

5 Analysis of the Algorithm

We first need some notation. Let Z be the value of the semidefinite program. For $a, b \in \{1,2,3\}$, let S_{ab} be the set of edges with one endpoint in S_a and the other endpoint in S_b. For any set of edges D, let $Val(D) = \sum_{(i,j)\in D} z_{ij}$ be the total value of the set, and $Cov(D)$ be the number of edges in D covered by the set C in our algorithm. Let C_1, C_2, C_3 be the number of vertices chosen in S_1, S_2, S_3 respectively before step 7.

By step 7 of the algorithm we are sure that the algorithm produces a set C which covers at least k edges. In order to show that we get a good approximation ratio, we have to bound the size of C. We show that the number of vertices added in step 7 is relatively small with high probability, and that the total number of vertices added before step 7 is small enough. In order to show this, we first show that with high probability for each of the sets S_{ij}, $Cov(S_{ij})$ is not much smaller than $Val(S_{ij})$ before step 7. Thus, the total number of covered edges before step 7 is not much smaller than the sum of all the z_{ij}, which is at least k as ensured by our relaxation.

Clearly, we cover all the edges with one endpoint in S_1. We therefore have to show that we cover many edges in S_{22}, S_{23} and S_{33}.

ALGORITHM part-cover

Input: A graph $G = (V, E)$ with maximum degree d, $|E| = m$, and $|V| = n$; a number $k \leq m$ is also given.
Output: A subset $C \subseteq V$ which covers at least k edges.

1. Solve relaxation (2); let the x_i and z_{ij} be as in (3).
2. Let x be a value to be determined later; x will be $\Theta(\frac{\ln \ln d}{\ln d})$. Let $S_1 = \{i \mid v_0 \cdot v_i \geq x\}$, $S_2 = \{i \mid -2x \leq v_0 \cdot v_i < x\}$, $S_3 = V \setminus (S_1 \cup S_2)$.
3. $C \leftarrow S_1 \cup S_2$.
4. Find a (hopefully large) subset I of S_2, using the following procedure (see [8]). Let c be such that $N(c) = \Theta(\frac{\ln \ln d}{\ln d})$; $c \approx \sqrt{2 \ln \ln d}$ will be chosen precisely later. Choose an n-dimensional random vector r, and let $I = \{i \in S_2 \mid v_i \cdot r \geq c\}$. Let $C = C \setminus I$.
5. Renumber the vertices so that $I = \{1, 2, \ldots, |I|\}$. Let ϵ be a parameter to be determined later. Draw a vector $(X_1, \ldots, X_{|I|})$ from the distribution $\mathcal{D}'(|I|, (1-\epsilon, \ldots, 1-\epsilon))$. For each vertex $i \in I$, add i to C if $X_i = 1$.
6. If all edges with at least one end-point in S_3 have already been covered, skip to the next step; otherwise proceed as follows. Renumber the vertices so that $S_3 = \{1, 2, \ldots, |S_3|\}$. For each vertex $i \in S_3$, let $p_i = \frac{1+v_0 \cdot v_i}{1+2x} = \frac{2x_i}{1+2x}$. Let $(X_1, \ldots, X_{|S_3|})$ be a vector drawn from the distribution $\mathcal{D}'(|S_3|, (p_1, \ldots, p_{|S_3|}))$. Let $S = \{i \mid X_i = 1\}$. Repeat this step $\lceil 4|E|/x \rceil$ times independently, and finally choose the set S which covers the maximum number of yet-uncovered edges with at least one endpoint in S_3. If this maximum number is zero, re-define S to be the set containing any one vertex of S_3 which covers at least one yet-uncovered edge. (Such a vertex must exist, for otherwise we would have skipped this step.) Add S to C.
7. If C does not cover at least k edges, arbitrarily add vertices to C, such that each vertex adds at least one new covered edge. When C covers at least k edges, stop and output C.

Fig. 1. Algorithm **part-cover** for finding a small vertex cover in a graph

For each $(i, j) \in S_{22}$, let $y_{ij} = 1 + v_i \cdot v_j$, $D_1 = \{(i, j) \mid y_{ij} > 4x\}$, $D_2 = S_{22} - D_1$, and $Y_k = \sum_{(i,j) \in D_k} y_{ij}$ for $k = 1, 2$. We first present a useful bound on the probability P_{ij} that edge $(i, j) \in S_{22}$ is not covered at the end of step 5. By Lemma 3 and by property (A3) of Section 3.2, it can be verified that

$$P_{ij} \leq \epsilon^2 N\left(\frac{2c}{\|v_i + v_j\|}\right) = \epsilon^2 N\left(\frac{\sqrt{2}c}{\sqrt{y_{ij}}}\right). \tag{4}$$

We let $\mathbf{E}[R]$ denote the expectation of a random variable R. We start by showing that with high probability we will cover many edges of D_1.

Lemma 4. *If $c \geq \sqrt{2}$, then the probability that $Val(D_1) \geq Cov(D_1)$ is at most $4\epsilon^2 N(c)$.*

Proof. Let $(i,j) \in D_1$. We have

$$1 - z_{ij} = \frac{1 - v_0 \cdot v_i - v_0 \cdot v_j + v_i \cdot v_j}{4} \geq \frac{y_{ij} - 2x}{4} \geq \frac{y_{ij}}{8}.$$

By applying Lemma 1, one can verify that the function $(8\epsilon^2/y) \cdot N\left(\frac{\sqrt{2}c}{\sqrt{y}}\right)$ is an increasing function of y whenever $0 \leq y \leq 2$ and $c \geq \sqrt{2}$. Therefore, by (4), the ratio $\frac{P_{ij}}{1-z_{ij}}$ is bounded by $4\epsilon^2 N(c)$, and we get

$$\mathbf{E}[|D_1| - Cov(D_1)] \leq 4\epsilon^2 N(c)(|D_1| - Val(D_1)).$$

Hence,

$$\Pr(Cov(D_1) \leq Val(D_1)) = \Pr\left(|D_1| - Cov(D_1) \geq |D_1| - Val(D_1)\right) \leq 4\epsilon^2 N(c),$$

by Markov's inequality.

Next we consider the set D_2. We show that almost all the edges of D_2 are covered before step 7.

Lemma 5. *Let γ be such that $2N(c)^{\frac{1}{2x}} < \gamma$. For every $\beta > 0$, the probability that $Cov(D_2) \leq |D_2| - \beta|D_2|$ after step 5 is at most $\frac{\epsilon^2\gamma}{\beta}$.*

Proof. By (4), we have

$$\mathbf{E}[|D_2| - Cov(D_2)] \leq \epsilon^2 \sum_{(i,j)\in D_2} N\left(\frac{\sqrt{2}c}{\sqrt{y_{ij}}}\right).$$

It is easy to see that $N\left(\frac{\sqrt{2}c}{\sqrt{y}}\right)$ is an increasing function of y, and therefore, since for every $(i,j) \in D_2$ $y_{ij} \leq 4x$, by Lemma 1 we get that

$$\mathbf{E}[|D_2| - Cov(D_2)] \leq \epsilon^2 |D_2| N\left(\frac{c}{\sqrt{2x}}\right) \leq 2\epsilon^2 |D_2| \sqrt{x} N(c)^{1/2x} \leq \gamma\epsilon^2 |D_2|.$$

By Markov's inequality, the probability that $|D_2| - Cov(D_2) > \beta|D_2|$ is at most $\frac{\epsilon^2\gamma}{\beta}$.

We now consider the set S_{23}. For each $i \in S_2$, let W_i be the indicator for the event that i was chosen in step 4. As in Figure 1, let X_i be the indicator for the event that i was chosen in step 5. The following lemma shows that after step 5, the expected number of edges covered in S_{23} is high.

Lemma 6. *Let W_i, X_i be as defined a few lines above. Let A be a random variable defined by*

$$A = \sum_{(i,j)\in S_{23}} (1 - W_i)(1 - X_i)(1 - \frac{2x_j}{1+x}).$$

After step 5, $\Pr(A \geq \frac{|S_{23}| - Val(S_{23})}{2}) \leq 12\epsilon N(c)$.

Proof. Let $(i,j) \in S_{23}$, where $i \in S_2, j \in S_3$. We first bound the expression $1 - z_{ij}$.

$$1 - z_{ij} = \frac{1 + v_i \cdot v_j - v_0 \cdot v_j - v_0 \cdot v_i}{4} \geq \frac{-v_0 \cdot v_i - v_0 \cdot v_j}{2} \geq \frac{-x + 1 - 2x_j}{2}.$$

We also have that the expectation $\mathbf{E}[A]$ of A equals $\sum_{(i,j)\in S_{23}} \epsilon N(c)(1 - \frac{2x_j}{1+x})$. Now, since $j \in S_3$, we have $2x_j \leq 1 - 2x$; for any such j, we can check that $1 - \frac{2x_j}{1+x} \leq 3(1 - x - 2x_j)$. So,

$$\begin{aligned}\mathbf{E}[A] &\leq \sum_{(i,j)\in S_{23}} 3\epsilon N(c)(-x + 1 - 2x_j) \\ &\leq 6\epsilon N(c) \sum_{(i,j)\in S_{23}} (1 - z_{ij}) = 6\epsilon N(c)(|S_{23}| - Val(S_{23})).\end{aligned}$$

Markov's inequality now completes the proof.

Notation. For $i = 1, 2, 3$, let $\Phi_i = \sum_{j \in S_i} x_j$.

Using Lemma 6, we next show that enough edges are covered from S_{23} and S_{33} in step 6.

Lemma 7. *There is an absolute constant $\delta > 0$ such that the following holds as long as $x \leq \delta$. Let A be as in the statement of Lemma 6. Fix the random choices in the steps up to and including step 5, conditional on the bound $A \leq \frac{|S_{23}| - Val(S_{23})}{2}$. Then, with probability at least $1 - 1/e$, both of the following hold at the end of step 6:*

(a) at most $(2 - 2x)\Phi_3 + 1$ elements of S_3 have been chosen; and
(b) $Cov(S_{23}) + Cov(S_{33}) \geq Val(S_{23}) + Val(S_{33})$.

This probability is only with respect to the random choices made in step 6.

Proof. If the test made at the beginning of step 6 led to the rest of the step being skipped, then items (a) and (b) hold trivially; so suppose the rest of the step was not skipped.

Item (a) is easy. Recall that step 6 repeats a "level-sets" trial $\lceil 4|E|/x \rceil$ times. By (A2'), we know that in each trial, the number of vertices chosen from S_3 is at most $\Phi_3 \cdot \frac{2}{1+2x} + 1 \leq \Phi_3(2 - 2x) + 1$, with probability one. Also, if we do re-define S to be an appropriate singleton set in step 6, item (a) still holds with probability one. Thus, we only need to focus on showing that item (b) holds with probability at least $1 - 1/e$, which we do now.

Note that A is now deterministic. In each of the $\lceil 4|E|/x \rceil$ "level-sets" trials, $\mathbf{E}[|S_{23}| - Cov(S_{23})] = A$. Consider one such trial. By the assumption of the lemma,

$$\mathbf{E}[|S_{23}| - Cov(S_{23})] \leq \frac{|S_{23}| - Val(S_{23})}{2}. \tag{5}$$

Consider the set S_{33}. Since the vertices chosen in step 6 are drawn from a level-set distribution, for every two vertices $i, j \in S_3$, the probability that neither of them will be chosen in S is at most $(1-p_i)(1-p_j)$; this follows from property (A3) of Section 3.2. Thus,

$$
\begin{aligned}
\mathbf{E}[Cov(S_{33})] &\geq \sum_{(i,j)\in S_{33}} (1-(1-p_i)(1-p_j)) \\
&= \sum_{(i,j)\in S_{33}} (\frac{2x_i+2x_j}{1+x} - \frac{4x_ix_j}{(1+x)^2}) \\
&\geq \sum_{(i,j)\in S_{33}} ((x_i+x_j)(\frac{2}{1+x} - \frac{x_i+x_j}{(1+x)^2})) \\
&\geq \sum_{(i,j)\in S_{33}} z_{ij} \cdot \frac{1+(2+2)x}{(1+x)^2} && (6) \\
&\geq (1+x)Val(S_{33}). && (7)
\end{aligned}
$$

Inequality (6) follows from the fact that $z_{ij} \leq x_i + x_j$ (see the last constraint of the semidefinite program), and since $i,j \in S_3$ implies that $x_i + x_j \leq 1-2x$. Bound (7) holds for all x small enough, since $1+4x \geq (1+x)^3$ holds for all small enough positive x.

The definitions of S_2 and S_3 imply that

$$Val(S_{23}) \leq \sum_{(i,j)\in S_{23}} (x_i+x_j) \leq \sum_{(i,j)\in S_{23}} (\frac{1+x}{2} + \frac{1-2x}{2}) = (1-x/2)\cdot |S_{23}|.$$

Thus, bounds (5) and (7) can be used to show that $\mathbf{E}[R] \geq (1+x/4)\cdot a$, where $R = Cov(S_{23}) + Cov(S_{33})$ and $a = Val(S_{23}) + Val(S_{33})$ for notational convenience. We wish to show that at the end of step 6,

$$\Pr(R \geq a) \geq 1 - 1/e. \tag{8}$$

This certainly holds if $a \leq 1$, since we ensure in step 6 that at least one yet-uncovered edge is chosen; so, suppose $a > 1$. Note that $R \leq |E|$ always; so by applying Markov's inequality to the random variable $|E| - R$, we get that in each trial of step 6,

$$
\begin{aligned}
\Pr(R < a) &\leq \frac{|E| - (1+x/4)\cdot a}{|E| - a} \\
&= 1 - (x/4)\cdot \frac{a}{|E|-a} \\
&\leq 1 - (x/4)\cdot \frac{1}{|E|}
\end{aligned}
$$

since $a > 1$. Thus, the probability that none of the $\lceil 4|E|/x \rceil$ trials succeeds in achieving "$R \geq a$" is at most

$$\left(1 - (x/4)\cdot\frac{1}{|E|}\right)^{\lceil 4|E|/x\rceil} \leq 1/e;$$

this proves (8).

We now prove the main theorem. In the theorems below, we content ourselves with showing that an algorithm produces a satisfactory solution with probability $\Omega(1/\log n)$; repeating the algorithm $a \log n$ times for a large enough constant a, will clearly lead to a satisfactory solution with high probability. Also, our results below have an extra "+2" term in the objective function. After Corollary 1, we discuss how to remove this extra term. We will let "$o(1)$" denote any function of d alone that goes to zero as d increases; different usages of such a term can correspond to different choices of such functions.

Theorem 1. *Let $G = (V, E)$ be a graph with maximal degree d. For sufficiently large d, with probability at least $\frac{1}{\ln d}$, algorithm* **part-cover** *will produce a k-cover of size at most $(2 - \Omega(\frac{\ln \ln d}{\ln d}))OPT + 2$, where OPT is the size of a minimal k-cover.*

Proof. We now choose our parameters; the choice of c is motivated by our aim to have $|N(c) - \frac{48 \ln \ln d}{\ln d}| = o(\frac{\ln \ln d}{\ln d})$. We choose:

$$c = \sqrt{2 \ln \left(\frac{\ln d}{96\sqrt{\pi}(\ln \ln d)^{3/2}} \right)};\; x = \frac{\ln \ln d}{6 \ln d};\; \text{and } \epsilon = \frac{1}{48}. \tag{9}$$

One can verify that the following properties hold for d large enough (to verify (10), we may use Lemma 1):

$$|N(c) - \frac{48 \ln \ln d}{\ln d}| = o(\frac{\ln \ln d}{\ln d}); \tag{10}$$

$$2N(c)^{1/2x} \leq \frac{2}{d^{3-o(1)}}; \tag{11}$$

$$(1 - 1/e) \cdot (N(c)/2 - 4\epsilon^2 N(c) - \frac{2\epsilon^2}{d} - 12\epsilon N(c)) \geq \frac{1}{\ln d}. \tag{12}$$

Recall that C_1, C_2, C_3 are the sizes of C intersecting S_1, S_2 and S_3 respectively. Since for each $i \in S_1$, $x_i \geq \frac{1+x}{2}$, we have that $C_1 \leq \Phi_1 \cdot \frac{2}{1+x} = (2 - 2x + O(x^2))\Phi_1$. Next, the expectation of C_2 after step 4 is $|S_2|(1 - N(c))$. Therefore, by applying Markov's inequality, the probability that $C_2 \geq (1 - \frac{N(c)}{2})|S_2|$ after step 4 is at most $p_1 = \frac{1-N(c)}{1-N(c)/2} \leq 1 - N(c)/2$. Thus, we can use property (A2') of the level-sets distribution to see that after step 5, $C_2 \leq |S_2|(1 - \epsilon N(c)/2) + 1$ holds with probability at least $1 - p_1$.

It is easy to see that the precondition "$c \geq \sqrt{2}$" of Lemma 4 and the precondition "x small enough" of Lemma 7 hold for all d sufficiently large. Since $\sum_{i,j=1,2,3} Val(S_{ij}) \geq k$, then by (11) and Lemmas 4, 5, 6 and 7, the following holds just before executing step 7. The probability that: (i) the number of covered edges is at least $k - |D_2|/d^{2-o(1)}$ (for some specific choice of the "$o(1)$" term) *and* (ii) $C_2 \leq |S_2|(1 - \epsilon N(c)/2) + 1$, is at least

$$p_2 = (1 - p_1 - 4\epsilon^2 N(c) - \frac{2\epsilon^2}{d} - 12\epsilon N(c)) \cdot (1 - 1/e).$$

Since the maximal degree of G is d, we have $|D_2| \leq |S_2|d$; thus, with probability at least p_2, we add in step 7 at most $\frac{|S_2|}{d^{1-o(1)}}$ vertices to C_2. Hence, the total number of vertices chosen after the algorithm terminates is at most

$$(2 - 2x + O(x^2))\Phi_1 + |S_2|(1 - \epsilon N(c)/2 + \frac{1}{d^{1-o(1)}}) + 1 + (2 - 2x)\Phi_3 + 1$$

with probability at least p_2. On the other hand, $\Phi_2 \geq |S_2|\frac{1-2x}{2}$. Thus, the algorithm finds a k-cover C, such that

$$\begin{aligned} |C| &\leq \max\{2 - 2x + O(x^2), 2 - (1 - o(1)) \cdot (\epsilon N(c) - 4x)\}Z + 2 \\ &\leq (2 - \frac{\ln \ln d}{(3 + o(1)) \ln d})OPT + 2 \end{aligned}$$

with probability at least p_2, for sufficiently large d; by (12), this probability is at least $\frac{1}{\ln d}$.

6 Extensions of the Analysis to Other Cases

The proof of Theorem 1 is based on the proper choice of c, x and ϵ. For graphs with bounded degree d, the choice of the parameters was given in the proof of the theorem. In this section we show how to choose the corresponding parameters in other graphs.

A p-claw in a graph is an *induced* star with $p + 1$ vertices (the center of the star and its p neighbors). A p-claw-free graph is a graph which does not contain any p-claw. Theorem 2 is motivated by a similar result from [5].

Theorem 2. *Let $G = (V, E)$ be a p-claw-free graph. For a proper choice of c, x and ϵ, algorithm* **part-cover** *will produce a k-cover C of size at most*

$$|C| \leq (2 - \Omega(\frac{\ln \ln p}{\ln p}))OPT + 2,$$

with probability at least $\Omega(\frac{1}{\ln p})$.

Proof. We choose our parameters as in (9), with d replaced by p. We claim that in D_2 there are no triangles. Assume that v_1, v_2, v_3 are vectors corresponding to a triangle in D_2. Since $v_i \cdot v_j \leq -1 + 4x$ for every $i \neq j, i, j \in \{1, 2, 3\}$, we get that

$$v_1 \cdot v_2 + v_1 \cdot v_3 + v_1 \cdot v_2 \leq -3 + 4x < -1,$$

which contradicts the constraints of the semidefinite program.

Since D_2 does not contain any triangle, and since it is a p-claw free graph, we get that the maximum degree in D_2 is $p - 1$. One can verify that in the proof of Theorem 1, we only used the property that G is a bounded degree graph when we bounded the number of edges in D_2. Thus, we can apply the same proof, replacing d by $p - 1$.

We now prove that when k decreases, the performance ratio of algorithm **part-cover** improves.

Theorem 3. *Let $G = (V, E)$ be a graph with a k-cover of size OPT. For a proper choice of c, x and ϵ, algorithm* **part-cover** *will produce a k-cover C of size at most*

$$|C| \leq (2 - \Omega(\frac{\ln \ln OPT}{\ln OPT}))OPT + 2,$$

with probability at least $\Omega(\frac{1}{\ln OPT})$.

Proof. We choose our parameters as in (9), with d replaced by $3 \cdot OPT$. Since $\Phi_2 \approx \frac{|S_2|}{2}$, we get that $|S_2| \leq 3 \cdot OPT$. Therefore, the maximum degree in D_2 is at most $3 \cdot OPT$. We can now apply the proof of Theorem 1, by replacing d by $3 \cdot OPT$.

Since the optimal k-cover of G contains at most k vertices, we get the following corollary.

Corollary 1. *Let $G = (V, E)$ be a graph. For a proper choice of c, x and ϵ, algorithm* **part-cover** *will produce a k-cover C of size at most*

$$|C| \leq (2 - \Omega(\frac{\ln \ln k}{\ln k}))OPT + 2,$$

with probability at least $\Omega(\frac{1}{\ln k})$.

We now mention how the extra "+2" term can be removed from the objective function value from the statements of Theorems 1, 2 and 3, as well as from Corollary 1. First note that our results imply that $OPT \leq 2Z + 2$. Suppose we adopt the following strategy. Let c_0 be a sufficiently large constant. If $Z \leq c_0$, we find an optimal solution in polynomial time by exhaustive search, since OPT is bounded by a constant. Next if $Z \geq c_0$ and if c_0 has been chosen large enough, then $(2 - \Omega(\frac{\ln \ln OPT}{\ln OPT}))OPT + 2 = (2 - \Omega(\frac{\ln \ln OPT}{\ln OPT}))OPT$. Similarly, consider Theorem 1, where we show an upper bound of the form $(2 - f(d))OPT + 2$, where $f(d) = o(1)$. Again, if $Z \leq c_0$, we find an optimal solution in polynomial time by exhaustive search; if $Z \geq c_0$, we use Theorem 1. Now if $f(d) \cdot OPT \geq 3$, then $(2 - f(d))OPT + 2$ is of the form $(2 - \Omega(f(d)))OPT$. Otherwise if $f(d) \cdot OPT < 3$, showing an upper bound of $2 \cdot OPT - 3$ on the output, will suffice to show that the output is at most $(2 - f(d))OPT$. Such an upper bound of $2 \cdot OPT - 3$ on the output follows from Theorem 3, since $Z \geq c_0$ and c_0 is large enough. Similar remarks hold for Theorem 1 and Corollary 1.

7 Concluding Remarks

We have presented an improved approximation algorithm for partial vertex cover using semidefinite programming. Our algorithm involves a new rounding technique, which copes with a few different worst-case configurations of the vectors at the same time. Our technique combines rounding methods for semidefinite programming, with additional ingredients such as level-sets distributions. It would be interesting to find other applications of such distributions in the context of semidefinite programming. Finally, we raise the following open problem. Is there a better approximation algorithm for partial vertex cover when restricted to small k, and specifically, is it possible to find a $(2 - \Omega(1))$–approximation when $k = o(n)$?

References

1. R. Bar-Yehuda. Using homogeneous weights for approximating the partial cover problem. In *Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms, Baltimore, Maryland*, pages 71–75, 1999.
2. N. Bshouty and L. Burroughs. Massaging a linear programming solution to give a 2-approximation for a generalization of the vertex cover problem. In *Fifteenth Annual Symposium on the Theoretical Aspects of Computer Science*, pages 298–308, 1998.
3. W. Feller. *An Introduction to Probability Theory and its Applications.* John Wiley & Sons, New York, 3 edition, 1968.
4. R. Gandhi, S. Khuller, and A. Srinivasan. Approximation algorithms for partial covering problems. In *Proceedings of the 28th International Colloquium on Automata, Languages and Programming, Crete, Greece*, pages 225–236, 2001.
5. E. Halperin. Improved approximation algorithms for the vertex cover problem in graphs and hypergraphs. In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms, San Francisco, California*, pages 329–337, 2000.
6. E. Halperin, R. Nathaniel, and U. Zwick. Coloring k-colorable graphs using smaller palettes. In *Proceedings of the 12th Annual ACM-SIAM Symposium on Discrete Algorithms, Washington, D.C.*, pages 319–326, 2001.
7. D. S. Hochbaum. The t-vertex cover problem: Extending the half integrality framework with budget constraints. In *First International Workshop on Approximation Algorithms for Combinatorial Optimization Problems*, pages 111–122, 1998.
8. D. Karger, R. Motwani, and M. Sudan. Approximate graph coloring by semidefinite programming. *Journal of the ACM*, 45:246–265, 1998.
9. A. Rényi. *Probability Theory.* Elsevier/North-Holland, Amsterdam, London, New York, 1970.
10. A. Srinivasan. Distributions on level-sets with applications to approximation algorithms. In *Proceedings of the 42nd Annual IEEE Symposium on Foundations of Computer Science, Las Vegas, Nevada*, pages 588–597, 2001.