Vertex cover is hardest to approximate on regular graphs

Uriel Feige
Department of Computer Science and Applied Mathematics
The Weizmann Institute
Rehovot, Israel
uriel.feige@weizmann.ac.il

April 15, 2004

Abstract

Let $\rho > 1$ be an arbitrary constant. We show that if vertex cover can be approximated within a ratio ρ on regular graphs, then vertex cover can be approximated within the same ratio ρ on every graph.

1 Introduction

For a graph G on n vertices, a vertex cover is a set of vertices that covers (namely, is incident with) all edges. The problem of finding the minimum cardinality vertex cover is NP-hard. Let $\mathrm{vc}(G)$ denote the cardinality of the minimum vertex cover in graph G. An approximation algorithm for vertex cover is said to have approximation ratio ρ if on every graph G it finds a vertex cover of size at most $\rho \cdot \mathrm{vc}(G)$. Note that necessarily $\rho \geq 1$. All algorithms considered in this manuscript run in polynomial time. Rather simple algorithms approximate vertex cover within a ratio of 2. For example, one can iteratively pick a yet uncovered edge and place both its endpoints in the vertex cover. Despite much research effort, no approximation algorithm with approximation ratio strictly better than 2 is known for vertex cover. The only known improvements to the approximation ratio are in the low order terms, giving approximation ratios of $2 - \Theta(\log\log n/\log n)$ [2, 16, 12], which asymptotically approaches 2 when n grows (though may offer significant improvements for small values of n). It is known that even fairly sophisticated algorithms fail to approximate vertex cover within a ratio better than 2 (see [4], for example). It has been conjectured that there is no polynomial time algorithm that approximates vertex cover within a ratio better than 2.

Through a sequence of works (starting with [1], leading up to [6]) it was shown that it is NP-hard to approximate vertex cover within any ratio better than $10\sqrt{5} - 21 \simeq 1.36$. Some unproven conjectures (regarding the hardness of approximating the success probability of two prover games with the uniqueness property), if true, would allow to extend the NP-hardness of approximation results all the way up to 2 [14].

There are several NP-hard combinatorial optimization problems for which there are NP-hardness of approximation results that match the known approximation ratios for them. These problems are said to have an *approximation threshold*. For example, max-3SAT has a threshold at 7/8 [13], and the max coverage problem has a threshold at $1 - 1/e \simeq 0.632$ [7]. A plausible conjecture regarding vertex cover is that it has an approximation threshold at 2. The purpose of this manuscript is to investigate the compatibility of this conjecture with some observed properties of combinatorial optimization problems that are known to have a threshold of approximation. This relates to the notion of a *core*, which will be discussed shortly. More information about this notion can be found in [8].

1.1 The core of an approximation problem

The notion of a core of an approximation problem was suggested as a result of trying to find unifying themes in the currently known threshold of approximation results. Currently, there is no formal definition of this notion, but rather an intuitive understanding of what is meant by this notion. To understand this notion, one needs first to distinguish between approximation algorithms and estimation algorithms. An approximation algorithm for a combinatorial optimization problem is required to find a feasible solution, and then its quality (namely, approximation ratio) is measured as the ratio between the value of the solution that it finds and the value of the optimal solution. An estimation algorithm is not required to output a feasible solution. It is only required to output two numbers l (for "low") and h (for "high") such that necessarily the value of the optimal solution is sandwiched between l and h. Its quality (namely, estimation ratio) is measured as the ratio between h and l. Every approximation algorithm with approximation ratio ρ can trivially be modified to be an estimation algorithm with estimation ratio ρ. (The estimation algorithm simply outputs the value of the solution found by the approximation algorithm, and this value divided by ρ. The value of the optimal solution must be sandwiched between these two values.) Hence designing estimation algorithms is potentially easier than designing approximation algorithms. Indeed, there are some (though not many) combinatorial optimization problems for which the best estimation ratio known is strictly better than the best approximation ratio known.

It has been observed that NP-hardness of approximation results are typically also hardness of estimation results, within the same ratios. In particular, this holds for all problems (known to the author) for which a threshold of approximation has been established. Hence the thresholds of approximation are also thresholds of estimation.

We now reach the notion of a core of an NP-hard combinatorial optimization problem. Informally, a subset of the instances of the problem forms a core if the following conditions hold:

1. It is "easy" to determine whether a particular instance belongs to the core or not.

2. For instances in the core, there is an estimation threshold.

3. The estimation algorithm that achieves the estimation threshold is of a "trivial nature".

As an example, consider the problem of max-3SAT. The input is a 3CNF formula, and the goal is to estimate the maximum number of clauses that can be simultaneously satisfied. The core for max-3SAT is the set of instances for which no clause contains two identical literals. The 7/8-hardness of approximation result of [13] can be seen to apply also for this core, and moreover, it is a hardness of estimation result. Moreover, there is a trivial estimation algorithm for core instances: count the number of clauses (let us denote this number by m) and output $l = 7m/8$ and $h = m$. The maximum number of satisfied clauses is always sandwiched between these two numbers for instances in the core. (This is not true for instances not in the core, as can be seen by the 3CNF formula in one variable x, containing the two clauses $(x \vee x \vee x)$ and $(\bar{x} \vee \bar{x} \vee \bar{x})$.)

In [8], cores are identified for several other combinatorial optimization problems that have an approximation threshold.

The motivation for the current research was to see if the conjecture that vertex cover has an approximation threshold at 2 is compatible with the hypothesis that all "natural" combinatorial optimization problems that have an approximation threshold also have a core. The main finding is positive, and informally says that if vertex cover has an approximation threshold at 2 then it also has a core. Specifically, the core instances would be the regular graphs.

1.2 Notation

For a graph $G(V, E)$ and a set of vertices $S \subset V$, $|S|$ denotes its cardinality, $G(S)$ denotes the subgraph of G induced on S, and $N(S)$ denotes that set of vertices of G that are adjacent to some vertex in S.

2 Vertex cover on regular graphs

A graph is d-regular if every vertex has degree exactly d. Every n-vertex d-regular graph has a vertex cover of size at most $dn/(d+1)$. This is an immediate consequence of the fact that it has an independent set of size at least $n/(d+1)$. (A naive greedy algorithm produces such an independent set. Moreover, if the graph is connected, then it produces an independent set of size at least $(n-1)/d$.) The complement of an independent set is a vertex cover. As d grows, this upper bound on the size of the minimum vertex cover tends to n. As to a lower bound, every vertex cover in a regular graph is of size at least $n/2$, as the number of distinct edges covered by $n/2$ vertices is at most $dn/2$, which is the total number of edges in the graph.

Hence we propose the following trivial estimation algorithm for vertex cover on regular graph. Output $l = n/2$ and $h = n$. The size of the minimum vertex cover is necessarily sandwiched between l and h. The estimation ratio of this algorithm is 2. A trivial approximation algorithm with approximation ratio no worse than 2 is simply to output all vertices in the graph.

Our purpose is to show that if vertex cover has an approximation threshold (or more precisely, an estimation threshold) at 2, then regular graphs form a core for vertex cover. We already showed that two of the requirements for being a core hold. Namely, it is easy to recognize whether a graph is regular, and there is a trivial estimation algorithm on regular graphs that has an estimation ratio of 2. It remains to show that an estimation ratio better than 2 is NP-hard for regular graphs, assuming that it is NP-hard on some graphs. This is an immediate consequence of Theorem 1. Most of this manuscript is devoted to giving a detailed, self contained, proof of this theorem. In Section B in the Appendix we sketch a shorter version of the proof which uses as a "black-box" information that appears elsewhere.

Theorem 1 *If there is an approximation (estimation) algorithm for vertex cover with approximation ratio (estimation ratio) ρ on regular graphs, then there is an approximation (estimation) algorithm for vertex cover with approximation ratio (estimation ratio) ρ on every graph.*

Proof: We shall present the proof for the case of approximation algorithms. The proof for the case of estimation algorithms is similar.

Let us first note a simple lemma that will help us later overcome a minor technical difficulty (an extra +1 term that is introduced in Lemma 5).

Lemma 2 *Let $\rho > 1$ be an arbitrary constant. Assume that there is a polynomial time algorithm that for every graph G produces a vertex cover of size at most $\rho \cdot vc(G) + 1$. Then there is a polynomial time algorithm that for every graph G produces a vertex cover of size at most $\rho \cdot vc(G)$.*

Proof: Note that we can assume that $\rho > 4/3$, because approximating vertex cover within a ratio of 4/3 is NP-hard [6]. Pick three arbitrary disjoint edges in G. (If G does not have three disjoint edges, then $\mathrm{vc}(G) \leq 4$, and the minimum vertex cover can be found in polynomial time.) At least one endpoint from each edge belongs to the minimum vertex cover. This gives 8 possible ways of guessing three vertices that belong to the vertex cover. Try all of them, where for each choice simplify the graph by removing the three guessed vertices and all edges incident with them. Let

G' be a graph that results from a correct guess of three vertices in the vertex cover. Then $\mathrm{vc}(G') = \mathrm{vc}(G) - 3$. When the approximation algorithm finds a vertex cover in G', we can transform it to a vertex cover in G by adding in the three vertices. This vertex cover will have size at most:

$$(\rho \cdot \mathrm{vc}(G') + 1) + 3 = \rho(\mathrm{vc}(G) - 3) + 4 < \rho \cdot \mathrm{vc}(G)$$

where we have used the assumption that $\rho > 4/3$. □

Let G be an arbitrary graph on n vertices in which we wish to approximate vertex cover within a ratio of ρ. We shall transform it into a regular graph in three steps. The first step will give nonnegative weights to the edges so that the graph becomes regular. The second step duplicates vertices, and replaces every weighted edge by a number of unweighted edges, where this number is proportional to the weight of the edge. Some irregularities remain in the graph due to edges that had weight 0, and this is handled in a final step that adds to the graph a small number of vertices. Lemma 2 is used in order to undo the effect of this last step.

A *perfect fractional matching* is an assignment of nonnegative weights to the edges of the graph, such that the sum of weights of the edges incident with every vertex is exactly 1. Ideally, for the first step of our transformation, we would like to find a perfect fractional matching in G. However, not every graph has a perfect fractional matching, as can be seen by considering a path with two edges. To get over this problem, we appeal to the rich theory known about matchings (see for example [15, 20], or the survey [19], and references therein). It is known that if a graph has a perfect fractional matching then it has a so-called "perfect uncapacitated 2-matching" (which we call a "2-matching" for short), which is a perfect fractional matching in which the weight of all edges is either 0, 1/2 or 1. In particular, this implies that every regular graph has a 2-matching. A 2-matching decomposes into a set of disjoint odd cycles and isolated edges. (If it has an even cycle of weight 1/2, this cycle can be replaced by a matching of weight 1 on half the edges of the cycle.)

Definition 1 *A* 2-matching *in a graph is an edge induced subgraph that spans all vertices, and whose connected components are either odd cycles or edges.*

A 2-matching can be found in polynomial time (if it exists). There is a characterization for graphs that do not have a 2-matching. They necessarily have an independent set S with $|N(S)| < |S|$. The connection between 2-matchings and independent sets (and hence also vertex cover) that we will use is summarized in the following theorem.

Theorem 3 *Let G be an arbitrary graph. Then there is a polynomial time algorithm that partitions the vertex set of G into three sets, V_1, V_0 and $V_{1/2}$ with the following properties.*

1. *V_0 is an independent set. V_1 is the set of all neighbors of V_0. $|V_0| > |V_1|$.*

2. *Every minimum vertex cover of G contains all vertices of V_1 and no vertex from V_0.*

3. *For every vertex cover of $G(V_{1/2})$, its union with V_1 is a vertex cover of G.*

4. *$G(V_{1/2})$ has a 2-matching, which is produced by the polynomial time algorithm.*

Many theorems of a nature similar to Theorem 3 are known. In particular, Nemhauser and Trotter [17] show that the optimal solutions of the fractional vertex cover are half-integral, and that a minimum vertex cover (though not necessarily every minimum vertex cover, a distinction that is insignificant in our context) agrees with the values that are assigned to those vertices that receive integral values. Moreover, it can be shown that the subgraph induced on vertices that receive value $1/2$ in the optimal fractional vertex cover has a 2-matching, and conversely, every graph with a 2-matching has an optimal fractional vertex cover in which all vertices have value $1/2$. The proof of this last fact is facilitated by the fact that fractional vertex covers and fractional matchings are linear programming duals of each other. More information on the connection between 2-matchings and half-integral solutions for fractional vertex cover appears in [18]. A full proof of Theorem 3 may well have been published previously, but for completeness, we sketch a proof in Section A in the appendix. Using Theorem 3, we proceed with the proof of Theorem 1.

Given a graph G, partition its vertices into the sets V_1, V_0 and $V_{1/2}$, as given in Theorem 3. Clearly, an approximation of $\mathrm{vc}(G(V_{1/2}))$ within a ratio ρ implies an approximation of $\mathrm{vc}(G)$ within a ratio at least as good as ρ (by adding in the vertices of V_1). Hence it suffices to approximate $\mathrm{vc}(G(V_{1/2}))$. For simplicity of notation, we rename $G(V_{1/2})$ as G, and let n denote the number of its vertices. The above discussion is summarized in the following corollary.

Corollary 4 *In designing approximation algorithms for vertex cover, one may assume without loss of generality that the input graph G has a 2-matching. In particular, $vc(G) \geq n/2$.*

Now let G be an n vertex graph that contains a 2-matching in which we want to approximate vertex cover within a ratio of ρ. Our plan is to transform G into a regular graph in a way that essentially preserves the size of the minimum vertex cover (up to low order terms, and multiplication by a known integer k). For this, let as establish some notation. The set of isolated edges in the 2-matching will be called M, and the set of edges on the odd cycles of the 2-matching will be called C. Let Δ denote the maximum degree in G. Let d be the smallest odd integer satisfying $d \geq \Delta + 2$. Let D denote the total "degree deficiency" of G with respect to d, namely:

$$D = \sum_{v \in V} (d - d_v)$$

where d_v is the degree of vertex V. Observe that $d \leq n+2 \leq 2n \leq D \leq (n+1)n$, and both d and D are computable in polynomial time. Choose $k = 4D+1$. We perform a sequence of transformations, starting at G, going through G_k and $G_{k,M}$, and ending in G_R which is a regular graph.

- **The graph** G_k. The graph G_k is simply composed of k disjoint copies of the graph G, numbered from 0 to $k-1$. Clearly, its number of vertices is kn, and $\mathrm{vc}(G_k) = k \cdot \mathrm{vc}(G)$. Moreover, the relation between the sizes of the respective vertex covers is algorithmic. In particular, if one finds a vertex cover of size t in G_k, then it induces a vertex cover of size at most t/k in one of the copies of G.

- **The graph** $G_{k,M}$. It is obtained by adding $k-1$ edges to each vertex of G_k. For every matched edge $(u, v) \in M$, we add to G_k all edges of the form (u_i, v_j) for $0 \le i, j \le k-1$, $i \neq j$, where u_i denotes the vertex u in the ith copy of G, and v_j denotes the vertex v in the jth copy of G. There is no need to add the edges (u_i, v_i) because they are already present in G_k. For every odd cycle edge $(u, v) \in C$, we add to G_k all edges of the form (u_i, v_j) for $|i - j \bmod k| \le D$ and $i \neq j$. (Recall that $k = 4D + 1$.) Clearly, $\mathrm{vc}(G_{k,M}) \ge \mathrm{vc}(G_k)$, because all edges of G_k are present in $G_{k,m}$. In fact, we have equality, because a minimum vertex cover of G_k that is composed of identical minimum vertex covers in each of the k copies of G that compose G_k is also a vertex cover for $G_{k,M}$. Hence $\mathrm{vc}(G_{k,M}) = k \cdot \mathrm{vc}(G)$. Note intuitively that the graph $G_{k,M}$ is nearly regular. If the original graph G was nearly regular, then so is $G_{k,M}$. If the original graph G was highly irregular then its deficiency D was large, implying a large value for k. As all degrees in $G_{k,M}$ are between k and $k + \Delta - 1$, it is nearly regular when k is large.

- **The graph** G_R. Add a set R of D vertices to the graph $G_{k,M}$. Each vertex v of R reduces by one the deficiency of one vertex u in all its k copies in $G_{k,M}$, by placing the k edges (v, u_i) for all $0 \le i \le k-1$. As R has D vertices and the total deficiency in G was D, we have exactly the right number of vertices to make all deficiencies equal to 0, implying that all original vertices of $G_{k,M}$ have degree $d + k - 1$. To make the graph G_R regular we still need to handle the degrees of vertices in R which are k rather than $k + d - 1$. Observe that $d - 1$ is even (because d was chosen to be odd) and smaller than $D = |R|$. Hence we can add edges between vertices in R such that the subgraph induced on R is $(d-1)$-regular. (This can be done by iteratively adding an edge between the two vertices in R that currently have the lowest degree, until all vertices have degree $(d-1)$.) This completes the construction of R, which is regular of degree $k+d-1$. Note that $\mathrm{vc}(G_R) \ge \mathrm{vc}(G_{k,M})$ because G_R contains all edges of $G_{k,M}$. As an upper bound, $\mathrm{vc}(G_R) \le \mathrm{vc}(G_{k,M}) + D$ because every vertex cover of $G_{k,M}$ can be augmented to be a vertex cover of G_R by adding the vertices in R.

Lemma 5 *Using notation as above, every vertex cover in G_R of cardinality $\rho \cdot vc(G_R)$ can be transformed in polynomial time into a vertex cover in G of cardinality $\rho \cdot vc(G) + 1$.*

Proof: Suppose we find a vertex cover in G_R of size at most $\rho \cdot \mathrm{vc}(G_R)$. Using the fact that $\mathrm{vc}(G_R) \leq k \cdot \mathrm{vc}(G) + D$, it is also of size at most $\rho(k \cdot \mathrm{vc}(G) + D)$. At least in one of the k disjoint copies of G that belong to G_R it induces a vertex cover of size at most k times as small, namely at most $\rho \cdot \mathrm{vc}(G) + \rho \cdot D/k$. Using the facts that $k = 4D+1$ and $\rho \leq 2$, this means that we find in one of the copies of G a vertex cover of size at most $\rho \cdot \mathrm{vc}(G) + 1$. $\square$

Combining Lemma 5 and Lemma 2, we are done.

Summarizing, we have shown that from every graph G with n vertices (now G is the original graph, before the renaming) we can construct in polynomial time a regular graph G_R with $O(n^3)$ vertices such that if we can approximate vertex cover on G_R within a ratio of ρ, we can approximate vertex cover of G within a the same ratio ρ. This completes the proof of Theorem 1. $\square$

We remark that ρ in the statement of Theorem 1 is assumed to be a constant independent of the number of vertices n. If it is allowed to change as a function of n (e.g., to be of the form $2 - c \log\log n/\log n$, for some constant c, as the current best approximation ratios for vertex cover are), then our reduction to regular graphs may effect the terms that depend on n in the approximation ratio, because our reduction does not preserve the value of n.

3 Discussion

The results of this paper extend to show that if we can approximate vertex cover on regular graphs within a ratio of ρ, then we can also approximated the weighted vertex cover problem (where vertices have nonnegative weights and one seeks a vertex cover of minimum total weight) within the same ratio. This follows from the fact that weighted vertex cover can be approximated within the same ratio as vertex cover [5].

We have shown that if vertex cover has a threshold of approximation at 2, then this threshold already manifests itself for regular graphs. We would like to note here a combinatorial optimization problem related to vertex cover for which the approximation ratio on regular graphs is provably better than that on arbitrary graphs. This is the min-sum-vertex-cover problem, in which one needs to number the vertices in a graph from 1 to n, an edge is considered covered at a value which is the minimum of the two numbers given to its endpoints, and one seeks to minimize the sum of covering values over all edges. For more details, see [10]. Another problem of interest in this context is max-bisection, for which the approximation ratio known for regular graphs is better than that currently known for arbitrary graphs [9].

Let us discuss shortly a few known algorithmic approaches for approximating vertex cover on regular graphs. As indicated in the first paragraph of the introduction, every connected d-regular graph has an independent set of size at least $(n-1)/d$, which easily leads to an approximation algorithm for vertex cover of ratio $2 - 2/d$. Hence for small d, the approximation ratio is significantly better than 2. For larger values of d, an approximation ratio of $2 - (1 - o(1))2 \log\log d/\log d$ can be achieved using semidefinite programming [12]. As d grows, this approximation ratio converges to 2. A useful fact about d-regular graphs is that whenever they have an independent set of size αn for some $\alpha > 0$, their adjacency matrix necessarily has a negative eigenvalue of absolute value at least $\alpha d/(1-\alpha)$. (This result is due to A. Hoffman. For a proof, see [22], for example.) Hence the absence of such an eigenvalue, a fact that can be checked in polynomial time, excludes the existence of an independent set of size αn, and hence provides an estimation ratio of $1/(1-\alpha)$ for vertex cover. For most regular graphs, the most negative eigenvalue is of absolute value at most $O(\sqrt{d})$ [11], which implies an estimation ratio of $1+O(1/\sqrt{d})$ for vertex cover on these graphs. This ratio approaches 1 when d is large. Hence on most d-regular graphs (regardless of whether d is small or large), the value of the minimum vertex cover can be estimated within a factor significantly better than 2. The regular graph G_R that results from the reduction in Theorem 1 is rather special in this respect. Its degree is roughly k, and it has multiple negative eigenvalues of value below $-k$, regardless of the size of $\text{vc}(G_R)$. (Every edge in M gives rise to a k by k complete bipartite graph in G, which can be shown to imply an eigenvalue with value below $-k$, using the notion of Raleigh quotient.) Hence the eigenvalue approach by itself does not provide an estimation for $\text{vc}(G_R)$ within a ratio better than 2.

Acknowledgements

I thank Laci Lovasz for pointers to relevant references on 2-matchings, and Shimon Kogan on his help in Section B. This research was supported by The Israel Science Foundation (grant no. 263/02).

References

[1] S. Arora, C. Lund, R. Motwani, M. Sudan and M. Szegedy. “Proof verification and hardness of approximation problems”. *JACM 45(3): 501–555 (1998).*

[2] R. Bar-Yehuda and S. Even. ”A local-ratio theorem for approximating the weighted vertex cover problem”. *Annals of Discrete Mathematics* 25:27–45, 1985.

[3] C. Berge. “Regularizable graphs”. *Ann. Discrete Math.* 3 (1978) 11-19.

[4] M. Charikar. “On semidefinite programming relaxations for graph colorings and vertex cover”. In *Proc. 13th Symposium on Discrete Algorithms*, 616–620, 2002.

[5] P. Crescenzi, R. Silvestri and L. Trevisan. ”On weighted versus unweighted versions of combinatorial optimization problems”. *Information and Computation* 167(1):10–26, 2001.

[6] I. Dinur and S. Safra. “The importance of being biased”. In *Proc. 34th ACM Symposium on Theory of Computing*, 33–42, 2002.

[7] U. Feige. "A Threshold of $\ln n$ for Approximating Set Cover". *Journal of the ACM*, 45(4), 634–652, 1998.

[8] U. Feige. "Approximation Thresholds for Combinatorial Optimization Problems". In *Proceedings of the International Congress of Mathematicians, Beijing 2002, Vol. III, 649–658*, Higher Education Press.

[9] U. Feige, M. Karpinski, M. Langberg. "A Note on Approximating Max-Bisection on Regular Graphs". *Information Processing Letters* 79(4):181–188 (2001).

[10] U. Feige, L. Lovasz and P. Tetali. "Approximating Min-sum Set Cover". Submitted to *Algorithmica*. Preliminary version in *Approximation Algorithms for Combinatorial Optimization, proceedings of 5th International Workshop, Approx 2002*, LNCS 2462 Springer, 94–107, 2002.

[11] J. Friedman, J.Kahn and A. Szemeredi. "On the second eigenvalue in random regular graphs". In *Proceding of the 25th Annual ACM Symposium on Theory of Computing*, 587–598, 1989.

[12] E. Halperin. "Improved approximation algorithms for the vertex cover problem in graphs and hypergraphs". In *Proc. 11th Symposium on Discrete Algorithms*, 329–337, 2000.

[13] J. Hastad. "Some optimal inapproximability results". *JACM* 48(4): 798–859 (2001).

[14] S. Khot and O. Regev. "Vertex cover might be hard to approximate within $2-\epsilon$, *CCC*, 2003.

[15] L. Lovasz and M. Plummer. *Matching Theory.* Elsevier 1986.

[16] B. Monien and E. Speckenmeyer. "Ramsey numbers and an approximation algorithm for the vertex cover problem". *Acta Informatica* 22:115–123, 1985.

[17] G. Nemhauser and L. Trotter. "Vertex packings: Structural properties and algorithms." *Mathematical Programming*, 8:232–248, 1975.

[18] W. Pulleyblank. "Minimum node covers and 2-bicritical graphs". *Mathematical Programming* 17 (1979) 91–103.

[19] W. Pulleyblank. "Matchings and extentions". In *Handbook of Combinatorics*, Edited by R. Graham, M. Grotschel and L. Lovasz, Elsevier, 1995.

[20] E. Scheinerman and D. Ullman. *Fractional Graph Theory: A Rational Approach to the Theory of Graphs.* Wiley, 1997.

[21] L. Valiant and V. Vazirani. "NP is as easy as detecting unique solutions". *Theoretical Computer Science* 47:85–93, 1986.

[22] J. van Lint and R. Wilson. *A Course in Combinatorics.* Cambridge University Press, 1992.

A Finding a 2-matching in a graph

For completeness, we present a proof of Theorem 3.

Proof: A *partial 2-matching* of graph G is an edge induced subgraph whose connected components are edges and odd cycles, but it need not span all vertices. Given an arbitrary partial 2-matching P, we show a polynomial time algorithm *extend* that ends with one of the following two outcomes.

1. Either it finds a new 2-matching that spans more vertices than M does.

2. Or it finds a nonempty independent set of vertices V_0, such that all their neighboring vertices are in a set V_1, $|V_1| < |V_0|$, and the unique maximum independent set in $G(V_0 \bigcup V_1)$ is V_0. Hence V_0 is in every maximum independent set of G, and V_1 is in every minimum vertex cover.

Theorem 3 is easily proved by repeatedly applying algorithm *extend* until all vertices are classified as either being in V_0, or in V_1, or as participating in the partial 2-matching.

The input to algorithm *extend* is a graph G and a partial 2-matching P in G. We let M denote the set of edges in the partial matching, and C the set of odd cycles. We assume that all vertices that were classified in previous iterations of algorithm *extend* as belonging to V_0 or V_1 were removed from G.

A vertex not spanned by the partial 2-matching P is called *exposed.* The graph G contains at least one exposed vertex r (otherwise we are done). Make it the root of a so called *alternating tree* T. The tree is constructed as follows. Along every path in T starting from the root r, edges will alternate as not being in M and being in M. All vertices at even distance from r in T (including r itself) are called *outer vertices.* All vertices at odd distance from r are called *inner vertices.* The tree is constructed step by step, always by hanging an additional subgraph of G on an outer vertex. The types of extension steps of the tree are as follows (given in order of preference).

1. **Connect to exposed vertex.** If there is an edge connecting an outer vertex to an exposed vertex t (other than r) then we add this edge to the tree. Now the path from r to t along the tree is an *alternating path*, where edges alternate between not being in M and being in M. By swapping out of M all edges of M along the path, and swapping in all edges not in M along the path, the cardinality of M increases by 1, and thus algorithm *extend* ends with outcome 1 above.

2. **Connect to odd cycle.** If there is an edge connecting an outer vertex to a vertex t on an odd cycle $c \in C$ then we add this edge to the tree. The path from r to t along the tree is an *alternating path*, where edges alternate between not being in M and being in M. Swap out of M all edges of M along the path, and swap in all edges not in M along the path. In addition, remove vertex t from the odd cycle c, that now becomes a path with an even number of vertices, and hence contains a perfect matching. Add the edges of this matching to M, and remove c from C. We get a new partial 2-matching that spans one more vertex than P does, namely, r. Thus algorithm *extend* ends with outcome 1 above.

3. **Close odd cycle.** If there is an edge (u, v) connecting two outer vertices in T, use it to close an odd cycle composed of vertices in T. Add this odd cycle to C, and remove the edges belonging to M along this cycle from M. In addition, let s be the vertex on this odd cycle closest to r in the tree T. Along the path from r to s, swap between the edges in M and those not in M. We get a new partial 2-matching that spans one more vertex than P does, namely, r. Thus algorithm *extend* ends with outcome 1 above.

4. **Connect to matched edge.** If there is an edge connecting an outer vertex w to a vertex u that belongs to edge $(u, v) \in M$ (where $u \notin T$), add to T the edges (w, u) and (u, v). Note that u is an inner vertex of T and v is an outer vertex.

At some point, either an extension step of type 1, 2 or 3 is performed and the algorithm ends, or no more extension step is possible (because extension step 4 can be performed at most $|M|$ times). If the latter possibility happens, then the outer vertices of T form an independent set, and all their neighbors are inner vertices in T. The number of outer vertices is one more than the number of inner vertices because all extension operations add a pair of inner and outer vertices, and r is an outer vertex. Moreover, the set of outer vertices forms a unique maximum independent set in T. This follows from the fact that any independent set with more than half the vertices of T must contain r, not contain its neighbors, and so on. Hence algorithm *extend* ends with outcome 2 above.

This completes the proof of Theorem 3. $\square$

To avoid the possibility of misunderstanding, let us stress here that even though every vertex of V_1 belongs to every minimum vertex cover, it is not necessarily the case that V_1 contains all vertices that belong to every minimum vertex cover. Moreover, no polynomial time algorithm can have this stronger property (unless NP has random polynomial time algorithms), as otherwise we could find the minimum vertex cover in every graph that has a unique minimum vertex cover, a problem that is known to be NP-hard via randomized reductions [21].

B Regular multi-graphs

We sketch here an alternative proof of Theorem 1. This proof was developed in collaboration with Shimon Kogan.

Nemhauser and Trotter [17] show that one can find in polynomial time a partition of all vertices in a graph G into three sets, V_0, V_1 and $V_{1/2}$ with the following properties:

1. There is a minimum vertex cover that contains all vertices from V_1 and no vertex from V_0.

2. There are no edges between V_0 and $V_{1/2}$. Hence the union of V_1 and a vertex cover for $G(V_{1/2})$ is a vertex cover for G.

3. The unique fractional vertex cover for $G(V_{1/2})$ is the one that gives all vertices in $V_{1/2}$ value 1/2.

Berge [3] shows that the graph induced on $V_{1/2}$ above is *regularizable* (namely, edges can be added in parallel to existing edges so that the resulting multigraph becomes regular). For completeness (and to explicitly demonstrate that the regularization process can be performed in polynomial time), we sketch a proof for this fact.

Pulleyblank [18] shows that for $V_{1/2}$ as above and for every vertex $v \in V_{1/2}$, $G(V_{1/2} \setminus \{v\})$ has a 2-matching. ($G(V_{1/2} \setminus \{v\})$ is the subgraph induced on all vertices of $V_{1/2}$ except for v, and for the notion of 2-matching see Definition 1.) Consider an arbitrary deterministic polynomial time algorithm for finding a 2-matching (there are several such algorithms), and denote by M_v the 2-matching that it finds on $G(V_{1/2} \setminus \{v\})$. We can now deduce the following theorem.

Theorem 6 *The graph $G(V_{1/2})$ above can be changed in polynomial time into a regular multi-graph (a graph with parallel edges in which all vertices have the same degree) by replacing every one of the m edges of $G(V_{1/2})$ by at most $4m$ (and at least 1) parallel edges.*

Proof: Let d_v denote the degree of vertex v in $G(V_{1/2})$. Let $n(v,e)$ be equal to 0 if edge e is not in the 2-matching M_v, be equal to 2 if e is a matching edge in M_v, and equal to 1 if e is a cycle edge in M_v. Replace every edge e in $G(V_{1/2})$ by $2 + \sum_{v \in V_{1/2}} d_v \cdot n(v,e)$ parallel edges. Note that M_v for $v \neq u$ increases the degree of u by $2d_v$ The degree of u becomes $2d_u + \sum_{v \neq u} 2d_v = 4m$. $\square$

Every edge of $G(V_{1/2})$ is represented in the multi-graph, and hence vertex covers are preserved. This suffices to prove theorem 1 for multi-graphs rather than simple graphs. Moreover, this is done without increasing the number of vertices n. Note that similarly to the case of regular simple graphs, also for regular multi-graphs it holds that the minimum vertex cover contains half the vertices.

We can replace the regular multi-graph by a regular graph by making k^* disjoint copies of the original graph, where k^* is the maximum multiplicity of an edge in the multi-graph. The size of the minimum vertex cover increases by a factor of k^*. In addition, for every multi-edge with multiplicity k we add $k-1$ additional disjoint perfect matchings between the two groups of k^* vertices that correspond to its endpoints. This makes the resulting graph regular, without changing the size of the minimum vertex cover.

This completes the sketch of (the alternative) proof of Theorem 1.