A Kernel of Order $2k - c \log k$ for Vertex Cover

Michael Lampis[a]

[a] *Graduate Center, City University of New York, 365 5th Avenue, New York, NY*

Abstract

In a recent paper Soleimanfallah and Yeo proposed a kernelization algorithm for Vertex Cover which, for any fixed constant c, produces a kernel of order $2k-c$ in polynomial time. In this paper we show how their techniques can be extended to improve the produced kernel to order $2k - c \log k$, for any fixed constant c.

Keywords: Parameterized Complexity, Kernelization Algorithm, Vertex Cover

1. Introduction

A vertex cover of a graph $G(V, E)$ is a set $S \subseteq V$ such that all edges of E have at least one endpoint in S. The subject of this paper is the Vertex Cover problem which can be posed as follows: given a graph G and an integer k, does there exist a vertex cover of G with at most k vertices?

Vertex Cover is one of the most fundamental and widely studied graph-theoretic problems in theoretical computer science: it was one of Karp's original 21 NP-complete problems, it has received a lot of attention in the context of approximation algorithms and it is one of the flagship problems of the parameterized complexity community.

Here we investigate the problem from the parameterized point of view, and specifically from the point of view of kernelization algorithms. A kernelization algorithm is a polynomial-time algorithm which, given an instance of the problem (G, k) constructs an equivalent instance (G', k') with $k' \leq k$ and $|G'| \leq f(k)$ for some function f. Informally, kernelization algorithms are meant to capture the practical concept of data reduction and simplification rules and we would like to design such algorithms with $f(k)$ being as small as possible. We call $f(k)$ the size of the produced kernel. For more information on parameterized complexity and kernelization algorithms see [7, 11].

For Vertex Cover an algorithm that produces a kernel of size $2k$ has long been known (see [2]). It is strongly believed that this is asymptotically the best possible: Chen et al. [1] show that if there exists a kernelization algorithm that produces a kernel of size $(2 - \epsilon)k$ which is also a subgraph of the original graph then P=NP. Also, assuming the Unique Games Conjecture it is not possible

Email address: `mlampis@gc.cuny.edu` (Michael Lampis) to approximate Vertex Cover with a factor better than 2 in polynomial time ([8]). In related work it is shown that it is not possible to produce a kernel with $O(k^{2-\epsilon})$ edges unless the polynomial hierarchy collapses [5].

The above motivate an investigation of possible kernelization algorithms that would produce kernels of order $2k - g(k)$ for some function $g(k) = o(k)$. It is observed in [3] that a kernel of order $2k - 1$ is possible and more recently this was improved to a kernel of order $2k - c$, for any fixed integer constant c by Soleimanfallah and Yeo [13]. In this paper we continue this line of research, obtaining a kernel of order $2k - c \log k$.

The work of Soleimanfallah and Yeo relies heavily on classical work by Nemhauser and Trotter [10]. With that as a starting point Soleimanfallah and Yeo reduce the problem to that of deciding the satisfiability of a collection of roughly k^c 2-SAT instances. This essentially places a limit on the best result that can be achieved, since if c is some non-constant function of k, because k can be $\Theta(n)$ in some instances, we would get an algorithm requiring super-polynomial time.

Our main contribution here is to observe that rather than having to solve k^c instances of 2-SAT we can reduce the problem to that of deciding whether a 2-SAT formula can be made satisfiable by deleting at most c variables. Then, rather than using the trivial algorithm of checking all sets of variables of size c we rely on the seminal result of Razgon and O'Sullivan [12] to solve the problem in $O^*(15^c)$[1]. Thus, we obtain a polynomial-time algorithm not only when c is a fixed-constant but also when c is $\Theta(\log k)$.

2. Preliminaries

We will use $G(V, E)$ to denote the undirected input graph, n to denote its order and m to denote its size. If $V' \subseteq V$ we will denote by $G[V']$ the subgraph of G induced by the vertices of V'. We will write $\mathrm{vc}(G)$ to denote the size of a minimum vertex cover of the graph G.

Our approach will follow steps very similar to Soleimanfallah and Yeo [13]. First, we need the following theorem which simplifies the graph by producing a smaller instance induced by the vertices of a set V_0.

Theorem 1. *(Nemhauser and Trotter [10])*
There is an $O(m\sqrt{n})$ time algorithm which given a graph G computes two disjoint subsets of vertices of G, V_0, V_1, such that $\mathrm{vc}(G) = \mathrm{vc}(G[V_0]) + |V_1|$ *and* $\mathrm{vc}(G[V_0]) \geq |V_0|/2$.

Soleimanfallah and Yeo [13] also prove the following structural lemma:

Lemma 1. *(Soleimanfallah and Yeo [13])*
If G is a graph, V_0, V_1 as in Theorem 1 and M is a maximum matching of $G[V_0]$ then $\mathrm{vc}(G[V_0]) \geq |V_0| - |M|$.

[1]Faster algorithms for this problem have recently been proposed ([4]), but for our purposes here any singly-exponential FPT algorithm is sufficient

We will also need to use the following result:

Theorem 2. *(Razgon and O'Sullivan [12])*
Given a 2-SAT formula ϕ on n variables there exists an algorithm that decides if ϕ can be made satisfiable by deleting at most k clauses in time $O(15^k \cdot n^{O(1)})$.

In fact, we will need a slight variation of this result: we will rely on an algorithm of the same running time that decides if a 2-SAT formula can be made satisfiable by deleting at most k *variables*. The fact that Theorem 2 implies the existence of such an algorithm was already observed by Marx and Razgon [9]. We give the proof here for the sake of completeness.

Theorem 3. *(Marx and Razgon [9])*
Given a 2-SAT formula ϕ on n variables there exists an algorithm that decides if ϕ can be made satisfiable by selecting a set of at most k variables and deleting all the clauses that contain any of these variables, in time $O(15^k \cdot n^{O(1)})$.

Proof. Let $x_1, \dots, x_n$ be the variables of ϕ. We construct a new formula ϕ' as follows: we start with ϕ and introduce n new variables $y_1, \dots, y_n$. We replace every occurence of a literal $\neg x_i$ with the literal y_i. Finally, we add to the formula the clauses $(\neg x_i \vee \neg y_i)$ for all i (call these the consistency clauses).

We claim that ϕ can be made satisfiable by deleting k variables iff ϕ' can be made satisfiable by deleting k clauses. If ϕ can be made satisfiable by deleting some variables, for each such variable x_i we delete the clause $(\neg x_i \vee \neg y_i)$ from ϕ' and set both x_i and y_i to true, satisfying all the clauses where x_i appears in ϕ. For the remaining variables we use in ϕ' the same assignment as in ϕ and ϕ' is satisfied. If ϕ' can be made satisfiable by deleting k clauses, then it can be made satisfiable by deleting k of the added consistency clauses. To see this, suppose that ϕ' is made satisfiable by deleting one of the other clauses. Surely, both of its variables are set to false (otherwise the clause need not be deleted). We set one of them to true and delete its corresponding consistency clause. This cannot affect any of the other clauses, since all variables appear positively in them, and it cannot affect any other consistency clauses since they all contain different variables.

□

3. A Kernel of Order $2k - c\log k$

Here we give a proof of the improved kernel. This proof follows an outline similar to the proof of the main result of Soleimanfallah and Yeo [13] (in fact we try to use the same notation as much as possible), except that the final reduction is to Almost 2-SAT, rather than a large number of instances of 2-SAT.

Theorem 4. *For all $c > 0$ there exists a polynomial time algorithm that reduces any instance (G, k) of vertex cover to an instance of order $2k - c\log k$.*

Proof. First, apply the algorithm of Theorem 1 to the input graph to produce the two sets V_0, V_1. Set $k' = k - |V_1|$. It now follows that the instance $(G[V_0], k')$ has the same answer as the original instance. If $|V_0| \leq 2k' - c \log k' \leq 2k - c \log k$ then we have a kernel and we are done. So, in the remainder we will show that if $|V_0| > 2k' - c \log k'$ then we can compute the correct answer in polynomial time. Since $\text{vc}[G^*] \geq |V_0|/2$ we may also assume that $|V_0| \leq 2k'$, otherwise we can reject immediately.

We will use G^* to denote $G[V_0]$ and let $M = \{u_1v_1, u_2v_2, \ldots, u_{|M|}v_{|M|}\}$ be a maximum matching of G^* (such a matching M can be found in polynomial time [6]). By Lemma 1 we have that $\text{vc}(G^*) \geq |V_0| - |M|$. Thus, if $|M| \leq \frac{|V_0| - c \log k'}{2}$ then $\text{vc}(G^*) \geq \frac{|V_0| + c \log k'}{2}$. Because $|V_0| > 2k' - c \log k'$ this means that $\text{vc}(G^*) > k'$. So if $|M| \leq \frac{|V_0| - c \log k'}{2}$ we can immediately reject. Thus, from now on, we assume that $|M| > \frac{|V_0| - c \log k'}{2}$. We may also assume that $|M| \leq k'$, because otherwise we can again reject. Let $X = \{x_1, \ldots, x_{|X|}\}$ be the set of vertices not matched by M.

We will now construct a 2-SAT formula. It will consist of variables of two kinds, call them z_i and y_i. The informal meaning of the variables z_i will be to encode which of the two endpoints of an edge of M is included in a vertex cover, while the variables y_i will encode whether a vertex of X is included in the vertex cover.

The construction is as follows: for each edge $u_iv_i \in M$ we define a variable z_i and for each vertex $x_i \in X$ we define a variable y_i. For each edge of G^* connecting u_i to v_j we construct the clause $(z_i \vee \neg z_j)$. For each edge of the form u_iu_j we construct the clause $(z_i \vee z_j)$ and for each edge of the form v_iv_j we construct the clause $(\neg z_i \vee \neg z_j)$. Finally, for each edge of the form x_iu_j we construct the clause $(y_i \vee z_j)$ and for each edge of the form x_iv_j we construct the clause $(y_i \vee \neg z_j)$. We also add to the formula, for each vertex $x_i \in X$ the clause $(\neg y_i)$. This completes the construction; observe that we have added a clause for each non-matching edge of G^*, since there are no edges connecting two vertices of X (this would contradict the maximality of M).

It is not hard to see that the 2-SAT formula we have constructed is satisfiable iff there exists a vertex cover of size $|M|$. Furthermore, we will show a stronger connection: G^* has a vertex cover of size $|M| + l$ if there exist l variables in our formula such that deleting all the clauses which contain them makes the formula satisfiable. To see this, suppose that there exists a vertex cover of size $|M| + l$ and that it contains l_1 vertices of X and both of the endpoints of l_2 of the edges of M (clearly $l = l_1 + l_2$). We simply remove the l variables that correspond to these edges and vertices from the formula. We then set all the remaining y_i variables to false and give values to the z_i variables according to the informal meaning we described. This satisfies the formula, and it is not hard to see that the converse direction follows from a similar argument. More specifically, suppose that there exists a satisfying assignment for the formula after deleting l variables. We construct a vertex cover as follows: for each variable y_i that was deleted we include the corresponding vertex x_i in the cover; for each variable z_i that was deleted we include both u_i and v_i in the cover; finally for each of the non-deleted variables z_i we include in the cover either u_i or v_i, depending on whether z_i is set to true or false in the satisfying assignment respectively.

Since $|V_0| > 2k' - c\log k'$ and $|M| > \frac{|V_0| - c\log k'}{2}$ we have $|M| > k' - c\log k'$ therefore $k' - |M| < c\log k'$. Using the above reduction, we set $l = k' - |M|$ and ask if the produced formula can be made satisfiable by removing l variables. The algorithm of Theorem 3 gives an answer in $O^*(15^l) = k'^{O(c)}$, which is polynomial time for every fixed constant c. The 2-SAT algorithm will accept iff there is a vertex cover of size at most $|M| + l = k'$.

□

[1] J. Chen, H. Fernau, I. Kanj, and G. Xia. Parametric duality and kernelization: Lower bounds and upper bounds on kernel size. *SIAM Journal on Computing*, 37(4):1077–1106, 2008.

[2] J. Chen, I. Kanj, and W. Jia. Vertex Cover: Further observations and further improvements. *Journal of Algorithms*, 41(2):280–301, 2001.

[3] M. Chlebík and J. Chlebíková. Crown reductions for the minimum weighted vertex cover problem. *Discrete Applied Mathematics*, 156(3):292–312, 2008.

[4] M. Cygan, M. Pilipczuk, M. Pilipczuk, and J. O. Wojtaszczyk. On multiway cut parameterized above lower bounds. *CoRR*, abs/1107.1585, 2011.

[5] H. Dell and D. van Melkebeek. Satisfiability allows no nontrivial sparsification unless the polynomial-time hierarchy collapses. In L. J. Schulman, editor, *STOC*, pages 251–260. ACM, 2010.

[6] J. Edmonds. Paths, trees, and flowers. *Canadian Journal of Mathematics*, 17:449–467, 1965.

[7] J. Flum and M. Grohe. *Parameterized complexity theory*. Springer-Verlag New York Inc, 2006.

[8] S. Khot and O. Regev. Vertex cover might be hard to approximate to within 2-[epsilon]. *Journal of Computer and System Sciences*, 74(3):335–349, 2008.

[9] D. Marx and I. Razgon. Fixed-parameter tractability of multicut parameterized by the size of the cutset. In L. Fortnow and S. P. Vadhan, editors, *STOC*, pages 469–478. ACM, 2011.

[10] G. Nemhauser and L. Trotter. Vertex packings: structural properties and algorithms. *Mathematical Programming*, 8(1):232–248, 1975.

[11] R. Niedermeier. Invitation to Fixed-Parameter Algorithms. 2006.

[12] I. Razgon and B. O'Sullivan. Almost 2-SAT is fixed-parameter tractable. *Journal of Computer and System Sciences*, 75(8):435–450, 2009.

[13] A. Soleimanfallah and A. Yeo. A kernel of order 2k-c for Vertex Cover. *Discrete Mathematics*, 311(10-11):892–895, 2011.