Greedy domination on biclique-free graphs

Sebastian Siebertz

Humboldt-Universität zu Berlin

Abstract

The greedy algorithm for approximating dominating sets is a simple method that is known to compute an $(\ln n + 1)$ approximation of a minimum dominating set on any graph with n vertices. We show that a small modification of the greedy algorithm can be used to compute an $\mathcal{O}(t \cdot \ln k)$ approximation, where k is the size of a minimum dominating set, on graphs that exclude the complete bipartite graph $K_{t,t}$ as a subgraph.

Keywords: Dominating set problem, approximation algorithms, greedy algorithms, structural graph theory.

1. Introduction

A dominating set in an undirected and simple graph G is a set $D \subseteq V(G)$ such that every vertex $v \in V(G)$ lies either in D or has a neighbour in D. The Minimum Dominating Set problem takes a graph G as input and the objective is to find a minimum size dominating set of G. The corresponding decision problem is NP-hard [11] and this even holds in very restricted settings, e.g. on planar graphs of maximum degree 3 [7].

The following greedy algorithm computes an $H_n = \sum_{i=1}^{n} 1/i \leq \ln n + 1$ approximation of a minimum dominating set in an n-vertex graph [9, 13]. Starting with the empty dominating set D, the algorithm iteratively adds vertices to D according to the following greedy rule: in each round, choose the vertex $v \in V(G)$ that dominates the largest number of vertices which still need to be dominated. The greedy algorithm on general graphs is almost optimal: it is NP-hard to approximate a minimum dominating set to within factor $c \cdot \ln n$ for some constant $c > 0$ [14], and by a recent result it is even NP-hard to approximate a minimum dominating set to within factor $(1 - \epsilon) \cdot \ln n$ for every $\epsilon > 0$ [4].

[1] Full address: Sebastian Siebertz, Humboldt-Universität zu Berlin, Institut für Informatik, Chair for Algorithm Engeneering, email: `siebertz@informatik.hu-berlin.de`

[2] This work was supported by the National Science Centre of Poland via POLONEZ grant agreement UMO-2015/19/P/ST6/03998, which has received funding from the European Union's Horizon 2020 research and innovation programme (Marie Skłodowska-Curie grant agreement No. 665778).

November 3, 2018

On several restricted graph classes the Minimum Dominating Set problem can be approximated much better. For instance, the problem admits a polynomial-time approximation scheme (PTAS) on planar graphs [1] and, more generally, on graph classes with subexponential expansion [8]. It admits a constant factor approximation on classes of bounded arboricity [2] and an $\mathcal{O}(d \cdot \ln k)$ approximation (where k denotes the size of a minimum dominating set) on classes of VC-dimension d [3, 6].

While the above algorithms on restricted graph classes yield good approximations, they are computationally much more complex than the greedy algorithm. Unfortunately, the greedy algorithm does not provide any better approximation on these restricted graph classes than on general graphs (see for example Section 4 of [3] for an instance of the set cover problem, which can easily be transformed into a planar instance of the dominating set problem, where the greedy algorithm achieves only an $\Omega(\ln n)$ approximation). Jones et al. [10] showed how to slightly change the classical greedy algorithm to obtain a constant factor approximation algorithm on sparse graphs, more precisely, the algorithm computes a d^2 approximation of a minimum dominating set on any graph of degeneracy at most d.

Our results. We follow the approach of Jones et al. [10] and study small modifications of the greedy algorithm which lead to improved approximations on restricted graph classes. We denote the complete bipartite graph with i vertices on one side and j vertices on the other side by $K_{i,j}$.

We present a greedy algorithm which takes as input a graph G and an optional parameter $i \in \mathbb{N}$. If run with the integer parameter i and G excludes $K_{i,j}$ as a subgraph for some $j \geq 1$, then the algorithm computes an $\mathcal{O}\big(i^2 \cdot \ln k + i \cdot \ln j\big)$ approximation of a minimum dominating set of G (where k denotes the size of a minimum dominating set). If run without the integer parameter i, the algorithm outputs the largest subgraph $K_{t,t}$ that it found during its computation, as well as an $\mathcal{O}(t^2 \cdot \ln k + t \cdot \ln t) = \mathcal{O}(t^2 \cdot \ln k)$ approximation of a minimum dominating set. By running the classical greedy algorithm in parallel, the approximation ratios can be improved to $\mathcal{O}(i \cdot \ln k + \ln j)$, and $\mathcal{O}(t \cdot \ln k)$, respectively.

Based on a known hardness result for the set cover problem on families with intersection 1 it is easy to show that it is unlikely that polynomial time constant factor approximations exist even on $K_{3,3}$-free graphs.

Comparison to other algorithms. Every $K_{t,t}$-free graph has VC-dimension at most t, hence the algorithms of [3, 6] achieve $\mathcal{O}(t \cdot \ln k)$ approximations on the graphs we consider. The algorithm presented in [3] is based on finding ϵ-nets with respect to a weight function and a polynomial number of reweighting steps. The algorithm presented in [6] requires solving a linear program. Hence, even though these algorithms achieve the same approximation bounds as our modified greedy algorithm, our algorithm is much easier to implement and has much better running times. On the other hand, $K_{t,t}$-free-graphs are strictly more general than degenerate graphs. Hence, our algorithm is applicable to a more general class of graphs than the algorithm of Jones et al. [10].

2. The greedy algorithm on biclique-free graphs

We first consider the following greedy algorithm which takes as input an optional parameter $i \in \mathbb{N}$ and a graph G. We start by presenting how the algorithm works if the parameter i is given with the input.

We initialise $D_0 \coloneqq \emptyset$ and $A_0 \coloneqq V(G)$. The set D_0 denotes the initial dominating set and A_0 denotes the set of vertices that have to be dominated. The algorithm runs in rounds and in every round it makes a greedy choice on a few vertices to add to the dominating set, until no vertices remain to be dominated. Formally, in each round $m = 1, \dots,$ we construct a new set D_m which is obtained from D_{m-1} by adding at most $i-1$ vertices $v_1, \dots, v_\ell$. The set A_m is obtained from A_{m-1} by removing $v_1, \dots, v_\ell$ and their neighbours. We output the set D_m as a dominating set, when $A_m = \emptyset$.

Let us describe a round of the modified greedy algorithm. Assume that after round m we have constructed a partial dominating set D_m and vertices A_m remain to be dominated. We choose ℓ vertices $v_1, \dots, v_\ell$, $\ell < i$, as follows. We choose as v_1 an arbitrary vertex that dominates the largest number of vertices which still need to be dominated, i.e., a vertex which maximises $|N[v_1] \cap A_m|$. Here, $N[v]$ denotes the neighbourhood of a vertex v, including the vertex v. Let $B_1 \coloneqq (N[v_1] \cap A_m) \setminus \{v_1\}$. We continue to choose vertices $v_2, \dots, v_\ell$ inductively as follows. If the vertices $v_1, \dots, v_s$ and sets $B_1, \dots, B_s \subseteq V(G)$ have been defined, we choose the next vertex v_{s+1} as an arbitrary vertex not in $\{v_1, \dots, v_s\}$ that dominates the largest number of vertices of B_s, i.e., a vertex which maximises $|N[v_{s+1}] \cap B_s|$ and let $B_{s+1} \coloneqq (N[v_{s+1}] \cap B_s) \setminus \{v_{s+1}\}$. We terminate this round and add $v_1, \dots, v_\ell$ to D_{m+1} if either we have $\ell = i-1$, or $N[v] \cap B_\ell = \emptyset$ for each $v \in V(G) \setminus \{v_1, \dots, v_\ell\}$. We mark the vertices $v_1, \dots, v_\ell$ and their neighbours as dominated, i.e., we remove from the set A_m all vertices of $\bigcup_{1 \le m \le \ell} N[v_m]$ to obtain the set A_{m+1} and start the next round.

The crucial difference between the above modified greedy algorithm and the classical greedy algorithm is that the former is guaranteed to choose in every round m at least one vertex from *every* minimum dominating set for A_m, given that A_m is still large. This is made precise in the following lemma.

Lemma 1. *Let G be a graph which excludes $K_{i,j}$ as a subgraph. Let $A_m \subseteq V(G)$ be a set of vertices to be dominated and let M be a dominating set of A_m of size k in G. If $|A_m| \ge k^i \cdot (j+i)$, then the algorithm applied to A_m will find vertices $v_1, \dots, v_\ell$ with $M \cap \{v_1, \dots, v_\ell\} \ne \emptyset$.*

Proof. By assumption, A_m is dominated by the set M of size k. Hence there must exist a vertex $v_1 \in V(G)$ which dominates at least a $1/k$ fraction of A_m, that is, at least $k^{i-1} \cdot (j+i)$ vertices of A_m. Let $B_1 \coloneqq (N[v_1] \cap A_m) \setminus \{v_1\}$, hence $|B_1| \ge k^{i-1} \cdot (j+i) - 1 \ge k^{i-1} \cdot (j+i-1)$.

Assume $v_1 \notin M$. We repeat the same argument as above for B_1. Also B_1 is dominated by M of size k, hence there must exist a vertex $v_2 \in V(G)$ which dominates at least a $1/k$ fraction of B_1, that is, at least $k^{i-2} \cdot (j+i-1)$ vertices of B_1. Let $B_2 \coloneqq (N[v_2] \cap B_1) \setminus \{v_2\}$, hence $|B_2| \ge k^{i-2} \cdot (j+i-1) - 1 \ge$

$k^{i-2} \cdot (j+i-2)$. We repeat the argument for $v_2, v_3, \ldots, v_\ell$ and $B_2, B_3, \ldots, B_\ell$, each B_x for $0 \leq x \leq \ell < i$ (set $B_0 = A_m$) of size at least $k^{i-x} \cdot (j+i-x)$, ending with a set B_ℓ of size at least $k \cdot (j+1)$.

Hence, assuming that $v_1, \ldots, v_\ell \notin M$, we have $\ell = i-1$ and there must exist a vertex v with $|N[v] \cap B_\ell \setminus \{v\}| \geq j$. Fix any subset $B = \{w_1, \ldots, w_j\}$ of $N[v] \cap B_\ell \setminus \{v\}$ of size exactly j. Then the vertices $v, v_1, \ldots, v_{i-1}$ and the vertices $w_1, \ldots, w_j$ form a subgraph $K_{i,j}$, contradicting that such a subgraph does not exist in G. Hence, one of $v_1, \ldots, v_\ell$ must be contained in M. □

Hence, as long as it remains to dominate a large set A_m, the modified greedy algorithm makes an almost optimal choice. Once we are left with a small set A_m, it performs only slightly worse than the classical greedy algorithm.

Theorem 2. *If G is a graph which excludes $K_{i,j}$ as a subgraph, then the modified greedy algorithm called with parameter i computes an $\mathcal{O}(i^2 \cdot \ln k + i \cdot \ln j)$ approximation of a minimum dominating set of G, where k is the size of a minimum dominating set of G.*

Proof. Fix any minimum dominating set M of size k of G. By Lemma 1, as long as it remains to dominate a set of size at least $k^i \cdot (j+i)$, the modified greedy algorithm chooses in every round at least one vertex of M. Hence, when it remains to dominate a set A_m of size smaller than $k^i \cdot (j+i)$, the algorithm has chosen at most $i \cdot k$ vertices.

Once we have reached this situation, let $n \coloneqq |A_m| \leq k^i \cdot (j+i)$. We argue just as in the proof of Lemma 1 that there exists a vertex $v \in V(G)$ which dominates at least a $1/k$ fraction of A_m, that is, a subset of A_m of size at least n/k. The algorithm chooses such a vertex together with at most i other vertices which in the worst case dominate nothing else. Hence after the first round we are left to dominate at most $n_1 = n - n/k = n \cdot (1 - 1/k)$ vertices. In the second round, we find again a vertex which dominates at least a $1/k$ fraction of the remaining vertices, hence after the second round we are left to dominate at most $n_2 = n_1 - n_1/k = n_1 \cdot (1-1/k) = n \cdot (1-1/k)^2$ vertices. We repeat this argumentation and conclude that after executing x rounds of the algorithm it remains to dominate at most $n_x = n \cdot (1 - \frac{1}{k})^x$ elements. Let us determine for what value of x we have $n_x < 1$, in which case we have dominated all vertices.

We have $n_x \leq n \cdot (1-1/k)^x < n \cdot e^{-x/k}$, where the last inequality follows from the bound $1 - z < e^{-z}$, which holds for all $z > 0$. Thus, for $x \geq k \cdot \ln n$ we have $n_x < n \cdot e^{-\ln n} = 1$. We conclude that the algorithm terminates after at most $k \cdot \ln n$ steps, in particular, it computes a dominating set of size at most $i \cdot k \cdot \ln n$. Now, as $n \leq k^i \cdot (j+i)$, we have $\ln n \in \mathcal{O}(i \cdot \ln k + \ln j)$. Hence, in total the set has size at most $\mathcal{O}\big(i \cdot k + (i^2 \cdot \ln k + i \cdot \ln j) \cdot k\big) \in \mathcal{O}\big((i^2 \cdot \ln k + i \cdot \ln j) \cdot k\big)$. □

With slightly more computational effort we can compute an $\mathcal{O}(i \cdot \ln k + \ln j)$ approximation on $K_{i,j}$-free graphs (and an $\mathcal{O}(t \cdot \ln k)$ approximation on $K_{t,t}$-free graphs, respectively) as follows. For each of the sets $D_0, D_1, \ldots$ constructed in the course of the algorithm, run the standard greedy algorithm to extend it to a dominating set, and return the smallest of the sets obtained in this way.

Letting p be the first index such that D_p dominates all but at most $n = k^i \cdot (i+j)$ vertices of the graph, the above argument shows that $|D_p| \leq i \cdot k$. The standard greedy algorithm then adds at most $\ln n \cdot k \in \mathcal{O}((i \cdot \ln k + \ln j) \cdot k)$ further vertices to the dominating set, resulting in a dominating set of size $\mathcal{O}((i \cdot \ln k + \ln j) \cdot k)$.

We now modify the algorithm slightly to work without the parameter i. In each round let the algorithm choose elements $v_1, \ldots, v_\ell$, defining sets $B_1, \ldots, B_\ell$ in the above notation, until we do not find a vertex $v_{\ell+1}$ defining a set $B_{\ell+1}$ with $|B_{\ell+1}| \geq \ell + 1$ any more. Let $t = \ell + 1$ for the largest ℓ that was encountered in any round. Hence, the modified algorithm chooses at most $t - 1$ elements in every round. Observe that in this construction, when we are at step i and the corresponding set B_i has size at least j, $1 \leq i \leq \ell$, $j \geq 1$, then we have found a subgraph $K_{i,j}$. Hence, t is the least number such that the algorithm did not find $K_{t,t}$ as a subgraph and we can argue as above that the algorithm performs as if $K_{t,t}$ was excluded from G. We output $K_{t-1,t-1}$ as a witness for this performance guarantee.

Finally, note that the algorithm can be used to approximate the minimum size of a set which dominates a given subset S of vertices of the graph, by initializing $A_0 = S$ instead of $A_0 = V(G)$.

3. Hardness beyond degenerate graphs

By the result of Bansal and Umboh [2] one can compute a $3d$ approximation of a minimum dominating set on any d-degenerate graph. The approximation factor was improved to $2d + 1$ by Dvořák [5]. To the best of our knowledge, degenerate graphs are currently the most general graphs on which polynomial time constant factor approximation algorithms for the dominating set problem are known. It is easy to see that the existence of such algorithms on bi-clique free graphs, even on $K_{3,3}$-free graphs, is unlikely. This result is a simple consequence of the following result of Kumar et al. [12]. Given a family $\mathcal{F}$ of subsets of a set A, a *set cover* is a subset $\mathcal{G} \subseteq \mathcal{F}$ such that $\bigcup_{F \in \mathcal{G}} F = A$. The Minimum Set Cover problem is to find a minimum size set cover. The *intersection* of a set family $\mathcal{F}$ is the maximum size of the intersection of two sets from $\mathcal{F}$.

Theorem 3 (Kumar et al. [12]). *The* Minimum Set Cover *problem on set families of intersection* 1 *cannot be approximated to within a factor of* $c\frac{\log n}{\log \log n}$ *for some constant* c *in polynomial time unless for some constant* $\epsilon < 1/2$ *it holds that* $\mathrm{NP} \subseteq \mathrm{DTIME}(2^{n^{1-\epsilon}})$.

Now it is easy to derive the following theorem.

Theorem 4. *The* Minimum Dominating Set *problem on* $K_{3,3}$*-free graphs cannot be approximated to within a factor of* $c\frac{\log n}{\log \log n}$ *for some constant* c *in polynomial time unless for some constant* $\epsilon < 1/2$ *it holds that* $\mathrm{NP} \subseteq \mathrm{DTIME}(2^{n^{1-\epsilon}})$.

Proof. Let $\mathcal{F}$ be an instance of the set cover problem with intersection 1. Let $A = \bigcup_{F \in \mathcal{F}} F$. We compute in polynomial time an instance of the Minimum

Dominating Set problem on a graph G as follows. We let $V(G) = A \cup \mathcal{F} \cup \{x, y\}$, where x, y are new vertices that do not appear in A. We add all edges $\{u, F\}$ if $u \in F$, as well as all edges $\{x, F\}$ for $F \in \mathcal{F}$ and the edge $\{x, y\}$.

Now if $\mathcal{G} \subseteq \mathcal{F}$ is a feasible solution for the set cover instance, then $\mathcal{G}$ (as a subset of G) together with the vertex x is a dominating set for G of size at most $|\mathcal{G}| + 1$. Conversely, let D be a dominating set for G. We construct another dominating set X such that $|X| \leq |D|$ and $X \subseteq \mathcal{F} \cup \{x\}$. We simply replace each $u \in A$ by a neighbour $F \in \mathcal{F}$. Furthermore, if $y \in D$, we replace y by x. Observe that x or y must belong to D, as y must be dominated. Hence, in any case, $x \in X$. Now $\mathcal{G} \cap X$ is a set cover of size $|X| - 1$. Hence, the reduction preserves approximations.

Let us show that G excludes $K_{3,3}$ as a subgraph. Assume towards a contradiction that $K_{3,3} \subseteq G$. Then $K_{2,3} \subseteq G - x$. Since G is bipartite we find elements $a_1, a_2 \in A$ and $F_1, F_2 \in \mathcal{F}$ (as vertices of G) with $\{a_i, F_j\} \in E(G)$, $i, j \in \{1, 2\}$, which form a $K_{2,2}$ subgraph of this graph. By construction of G we have $|F_1 \cap F_2| \geq |\{a_1, a_2\}| = 2$, contradicting that $\mathcal{F}$ is a set cover instance with intersection 1.

Finally observe that the reduction is obviously polynomial time computable. $\square$

Acklowledgements. I thank Saket Saurabh for pointing me to the work of Jones et al. [10]. I thank the anonymous reviewers for their valuable comments, and in particular for pointing out that the modified greedy algorithm can be improved to an $\mathcal{O}(t \cdot \ln k)$ approximation by running the classical greedy algorithm in parallel.

References

[1] B. S. Baker. Approximation algorithms for NP-complete problems on planar graphs. *Journal of the ACM (JACM)*, 41(1):153–180, 1994.

[2] N. Bansal and S. W. Umboh. Tight approximation bounds for dominating set on graphs of bounded arboricity. *Information Processing Letters*, 122:21–24, 2017.

[3] H. Brönnimann and M. T. Goodrich. Almost optimal set covers in finite VC-dimension. *Discrete & Computational Geometry*, 14(4):463–479, 1995.

[4] I. Dinur and D. Steurer. Analytical approach to parallel repetition. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 624–633. ACM, 2014.

[5] Z. Dvořák. On distance r-dominating and 2r-independent sets in sparse graphs. *arXiv preprint arXiv:1710.10010*, 2017.

[6] G. Even, D. Rawitz, and S. M. Shahar. Hitting sets when the VC-dimension is small. *Information Processing Letters*, 95(2):358–362, 2005.

[7] M. R. Garey and D. S. Johnson. *Computers and intractability*, volume 29. WH Freeman, New York, 2002.

[8] S. Har-Peled and K. Quanrud. Approximation algorithms for polynomial-expansion and low-density graphs. *SIAM Journal on Computing*, 46(6):1712–1744, 2017.

[9] D. S. Johnson. Approximation algorithms for combinatorial problems. *Journal of computer and system sciences*, 9(3):256–278, 1974.

[10] M. Jones, D. Lokshtanov, M. Ramanujan, S. Saurabh, and O. Suchý. Parameterized complexity of directed Steiner tree on sparse graphs. In *European Symposium on Algorithms*, pages 671–682. Springer, 2013.

[11] R. M. Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer, 1972.

[12] V. A. Kumar, S. Arya, and H. Ramesh. Hardness of set cover with intersection 1. In *International Colloquium on Automata, Languages, and Programming*, pages 624–635. Springer, 2000.

[13] L. Lovász. On the ratio of optimal integral and fractional covers. *Discrete mathematics*, 13(4):383–390, 1975.

[14] R. Raz and S. Safra. A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 475–484. ACM, 1997.