Algorithmic Aspect of Minus Domination on Small-Degree Graphs

Jin-Yong Lin[1], Ching-Hao Liu[1(✉)], and Sheung-Hung Poon[2]

[1] Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan
yongdottw@hotmail.com, chinghao.liu@gmail.com
[2] School of Computing and Informatics, Institut Teknologi Brunei, Gadong, Brunei Darussalam
sheung.hung.poon@gmail.com

Abstract. Let $G = (V, E)$ be an undirected graph. A *minus dominating function* for G is a function $f : V \to \{-1, 0, +1\}$ such that for each vertex $v \in V$, the sum of the function values over the closed neighborhood of v is positive. The *weight* of a minus dominating function f for G, denoted by $w(f(V))$, is $\sum f(v)$ over all vertices $v \in V$. The *minus domination (MD) number* of G is the minimum weight for any minus dominating function for G. The *minus domination (MD) problem* asks for the minus dominating function which contributes the MD number. In this paper, we first show that the MD problem is $W[2]$-hard for general graphs. Then we show that the MD problem is NP-complete for subcubic bipartite planar graphs. We further show that the MD problem is APX-hard for graphs of maximum degree seven. Lastly, we present the first fixed-parameter algorithm for the MD problem on subcubic graphs, which runs in $O^*(2.3761^{5k})$ time, where k is the MD number of the graph.

1 Introduction

Let $G = (V, E)$ be an undirected graph. A *minus dominating function* for G is a function $f : V \to \{-1, 0, +1\}$ such that for each vertex $v \in V$, the sum of the function values over the closed neighborhood of v is positive, where the *closed neighborhood* of v is the set contains v and all neighbors of v. The *weight* of a minus dominating function f for G, denoted by $w(f(V))$, is $\sum f(v)$ over all vertices $v \in V$. The *minus domination (MD) number* of G, denoted by $\gamma^-(G)$, is the minimum weight for any minus dominating function for G. The *minus domination (MD) problem* asks for the minus dominating function which contributes the MD number.

According to [9], we can see that the function values $+1$ and -1 of the signed domination problem can be formulated as yes-no decisions for social networks. In a similar fashion, the function values $+1$, 0, and -1 of the MD problem can be formulated as yes-uncertain-no decisions for in social networks. In the following, we first mention related work on complexities. Dunbar et al. [4,5] first showed that the MD problem is NP-complete for bipartite graphs and for chordal graphs, and can be solved in linear time for trees. Then, Damaschke [2]

© Springer International Publishing Switzerland 2015
D. Xu et al. (Eds.): COCOON 2015, LNCS 9198, pp. 337–348, 2015.
DOI: 10.1007/978-3-319-21398-9_27

showed that the MD problem is NP-complete for planar graphs of maximum degree 4. They [2] also showed that for every fixed k there is a polynomial-time algorithm, which runs in time $O(3^{8k} \cdot n^{8k})$, for deciding whether a given graph G of maximum degree 4 satisfies $\gamma^-(G) \leq k$. Recently, Faria et al. [6] showed that the MD problem is NP-complete for splitgraphs, and is polynomial for graphs of bounded rankwidth and for strongly chordal graphs.

Next, we survey the related work on parameterized complexities. It is known that Downey et al. [3] showed that the domination problem is $W[2]$-complete. Then, Faria et al. [6] showed that the MD problem has no fixed-parameter algorithm, i.e., not in $W[0]$, unless $P = NP$. They also show that the MD problem is fixed-parameter tractable for planar graphs, when parameterized by the *size* of f, the number of vertices $x \in V$ with $f(x) = 1$, and for d-degenerate graphs, when parameterized by the size of f and by d.

Lastly, we survey the related work on approximation complexities for graphs of bounded degree. First, Alimonti and Kann [1] showed that the domination problem is APX-hard on subcubic graphs. Then, Damaschke [2] showed that the MD number cannot be approximated in polynomial time within a factor $1 + \epsilon$, for some $\epsilon > 0$, for graphs of maximum degree 4, unless $P = NP$.

Outline. We organize the rest of this paper as follows. In Section 2, we show that the MD problem is $W[2]$-hard for general graphs. In Section 3, we show that the MD problem is NP-complete for subcubic bipartite planar graphs. In Section 4, we show the MD problem is APX-hard for graphs of maximum degree 7. In Section 5, with a very involved analysis, we obtain the first FPT-algorithm for the MD problem on subcubic graphs G, which runs in time $O^*(2.3761^{5k})$, where k is the MD number of G.

2 $W[2]$-hardness for General Graphs

In this section, by reducing the domination problem to the MD problem, we show that the MD problem on general connected graphs is $W[2]$-hard. Thus we introduce domination problem as follows. A *dominating set* of a graph $G = (V, E)$ is a vertex set $D \subseteq V$ such that for each vertex $v \in V$, there exists at least one vertex in the closed neighborhood $N_G[v]$ belonging to D. In other words, we can label the vertices in V with $\{0, +1\}$ such that the closed neighborhood of each vertex is positive. The *domination number* of G, denoted by $\gamma(G)$ is the cardinality of the minimum dominating set of G. The *domination problem* asks for a dominating set which contributes the domination number $\gamma(G)$. Downey et al. [3] showed that the domination problem on general connected graphs is $W[2]$-complete.

In our reduction, we need to make use of a graph H as shown in Figure 1(a). If we label the vertices of H as shown in Figure 1(a), where there are four vertices labeled with $+1$ and five vertices labeled with -1, then the weight for such a minus domination function is -1. In the following lemma, we show that -1 is in fact the minimum weight which a legitimate minus dominating function for graph H can provide.

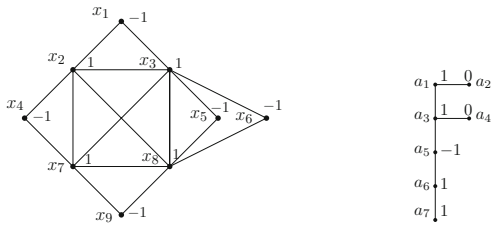


Fig. 1. (a) Graph H with weight -1. (b) A key with minimum weight 3.

Lemma 1. *The MD number of graph H shown in Figure 1(a) is at least -1.*

Proof. Suppose to the contrary that $\gamma^-(H) \leq -2$. Let f be a minus dominating function of H whose weight is not more than -2. Let P be the set of vertices labeled with $+1$ in H. If $\gamma^-(H) \leq -2$, then we have $|P| \leq 3$. It is clear that $|P| > 1$. If $|P| = 2$, it is easy to see that the rest vertices should label with 0 in H. Thus we have $|P| \neq 2$. If $|P| = 3$, then there are at most three vertices labeled with -1 in H. Thus we also have $|P| \neq 3$. With such a contradiction, we thus have $\gamma^-(H) \geq -1$. □

With this lemma, now we show the $W[2]$-hardness for the MD problem.

Theorem 1. *The MD problem is $W[2]$-hard for general connected graphs.*

Proof. We reduce the domination problem to our problem as follows. Given a graph $G = (V, E)$, we construct a new graph G' as described below. Initially we set G' to be G. Then we add each vertex v of G' a set of $\deg_G(v) + 1$ keys as shown in Figure 1(b), and connect v to a_7 of each keys in this set, where $\deg_G(v)$ is the degree of vertex v in G. Note that at most one of vertex in each key can be labeled with -1, and thus the weight for each key is greater than or equal to three. We let F be the set of newly added vertices. Next, we add $6m + 2n$ copies of graph H into graph G' and denote them by $H_1, H_2, \ldots, H_{6m+2n}$, respectively. Let vertex set $Y = V(H_1) \cup \ldots \cup V(H_{6m+2n})$. Then we take any previously added key, say ρ. Now, we connect the vertex a_1 of ρ to the corresponding vertex x_2 of each copy $H_i; i = \{1, \ldots, 6m + 2n\}$ of graph H. This completes the construction of the graph G'. Let V' be the set of vertices in G'. As G is a connected graph, graph G' is also a connected graph by our construction. In the following, we show that there is a domination set of size at most k in G if and only if there is a minus dominating function with weight at most k in G'.

Assume that there is a dominating set D of size at most k in G. Then we will show that there is a minus dominating function f with weight at most k of G'. We construct a function f, which labels vertices in F as Figure 1(b), vertices in D with value 0, vertices in $V \setminus D$ with value -1. We also label the corresponding vertices x_2, x_3, x_7, x_8 with value $+1$ and x_1, x_4, x_5, x_6, x_9 with value -1 for each

$H_i; i = \{1, \ldots, 6m + 2n\}$. Now, we claim that f is a minus dominating function of G' with weight at most k. Clearly, the sum weight of the closed neighborhood of any vertex in $F \cup Y$ is positive. Then we consider vertices in $V' \setminus (F \cup Y)$. For any vertex v in $V' \setminus (F \cup Y)$, since D is a dominating set, there is at least one vertex in $N_{G'}[v] \cap D$. Thus more than half of the vertices in $N_{G'}[v]$ are labeled with $+1$ in function f. Thus the weight for induced subgraph $G[N'_G[v]]$ is positive. Hence, f is a legitimate minus dominating function of G'. Since the weight of each H_i is at least -1 by Lemma 1, we obtain that

$$w(f(V')) \leq -(n-k) + (-1)(6m + 2n) + 3(2m + n) = k.$$

Conversely, we can also show that if f is a minus dominating function of G' with weight at most k, then there is a dominating set size at most k in G. The proof is omitted here. Hence, we complete the proof. □

3 NP-completeness for Subcubic Bipartite Planar Graphs

A graph is called *cubic* if its vertex degrees are degree three, and *subcubic* if its vertex degrees are at most three. It has been shown in [2] that the MD problem on planar graphs of maximum degree four is NP-complete. In this section, we show that the MD problem is NP-complete even for subcubic bipartite planar graphs. We leave open the question that whether the MD problem is NP-complete for cubic bipartite planar graphs. For showing our NP-completeness results, we make use of a lemma presented by Damaschke in [2].

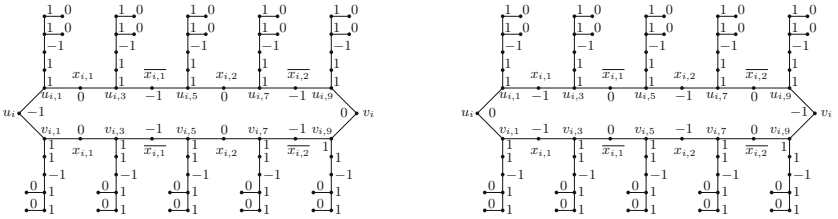
Lemma 2 (Lemma 3 of [2]). *In any graph, we consider a vertex x of degree 1 and the unique neighbor w of x. Then there is an optimal minus dominating function such that $f(x) = 0$ and $f(w) = 1$.*

Then we show the main NP-completeness theorem in the following.

Theorem 2. *The MD problem is NP-complete for subcubic bipartite planar graphs.*

Proof. Clearly, the problem is in NP. We reduce the planar 3SAT problem [7] to this problem. The input instance for the planar 3SAT problem is a set $\{x_1, x_2, \ldots, x_n\}$ of n variables and a Boolean expression with conjunctive normal form $\Phi = c_1 \wedge c_2 \wedge \ldots \wedge c_m$ of m clauses, where each clause consists of exactly three literals, such that the variable clause graph of the input instance is planar. The *planar 3SAT problem* asks for whether there exists a truth assignment to the variables so that the Boolean expression Φ is satisfied. We then describe our polynomial-time reduction as follows.

Variable Gadget. First, we construct the variable gadget V_i for a variable x_i. The variable gadget V_i for x_i is a circular linkage as shown in Figure 2(a). We connect $4m + 2$ keys (of Figure 1(b)) together as shown in Figure 2(a), where $2m + 1$ keys are connected as a *chain of keys* for the upper part of V_i, and the other $2m + 1$ keys are connected as a chain of keys for the lower part of V_i .



(a) (b)

Fig. 2. Gadget V_i for variable x_i: (a) $x_i = \text{TRUE}$; (b) $x_i = \text{FALSE}$. Note that vertices $x_{i,j}$ and $\overline{x_{i,j}}$ represents the positive and negative literals for x_i, respectively. They are used to connect to a clause gadget C_j.

The connection of the keys is performed by the introduction of a cycle of length $8m + 4$. See Figure 2(a). We call such a circular linkage a *cycle of keys*. See Figure 2(a). We let u_i and v_i be the leftmost and the rightmost endpoints of V_i. Then we let $u_i, u_{i,1}, u_{i,2}, \ldots, u_{i,4m+1}, v_i$ be the upper chain starting from u_i to v_i. Similarly, we let $u_i, v_{i,1}, v_{i,2}, \ldots, v_{i,4m+1}, v_i$ be the lower chain starting from u_i to v_i. The vertices $u_{i,2}, u_{i,4}, \ldots, u_{i,4m}$ and $v_{i,2}, v_{i,4}, \ldots, v_{i,4m}$ are used for connecting with clause gadgets. That is, $u_{i,2}, u_{i,6}, \ldots, u_{i,4m-2}$ and $v_{i,2}, v_{i,6}, \ldots, v_{i,4m-2}$ are parts for positive literals $x_{i,j}$, and $u_{i,4}, u_{i,8}, \ldots, u_{i,4m}$ and $v_{i,4}, v_{i,8}, \ldots, v_{i,4m}$ are parts for negative literals $\overline{x_{i,j}}$. The interior of the whole variable gadget can be duplicated to make a longer gadget so that there are enough ports on the variable gadget for connecting to the corresponding literal gadgets of the related clauses in the later context.

Next, we first describe the truth assignment of the optimal minus dominating function for a key. The labeling method in Figure 1(b) is optimal, whose weight is 3. There is another way of optimal labeling such that the lowest vertex of the key is labeled with value 0. Since the lowest vertex of a key is connected to the main body of variable gadget, it is always advantageous to use the labeling method shown in Figure 1(b).

Then we describe the truth assignment of the optimal minus dominating function for the whole variable gadget. To attain the assignment of minimum weight, the internal cycle of variable gadget V_i may be labeled as either the way in Figure 2(a) or the way in Figure 2(b). In either way, the sum of the weights $f(x)$ for $x \in V_i$ is $8m$ and such $f(x)$ is the minimum minus dominating function. We use the domination way in Figure 2(a) to represent that $x_i = \text{TRUE}$, and the other domination way in Figure 2(b) to represent that $x_i = \text{FALSE}$.

Clause Gadget. We use clause gadgets to connect to the variable gadgets directly and there is no link gadget. Now we are prepared to construct the clause gadget C_j for clause $c_j = x_i \vee x_k \vee x_\ell$ which contains 28 vertices, that is, $p_1, \ldots, p_8, q_1, \ldots, q_8, r_1, \ldots, r_8, s_1, s_2, s_3, t$, and 33 edges as shown in Figure 3(a). The vertices p_0, q_0 and r_0 lie in variable gadgets V_i, V_k and V_ℓ, respectively. If x_i is TRUE or FALSE, then p_0 connects to a vertex in V_i which represents $x_{i,j}$ or $\overline{x_{i,j}}$, respectively. Similarly, if x_k is TRUE or FALSE, then q_0 connects to a vertex in V_k which represents $x_{k,j}$ or $\overline{x_{k,j}}$, respectively. If x_ℓ is TRUE, then r_0 connects

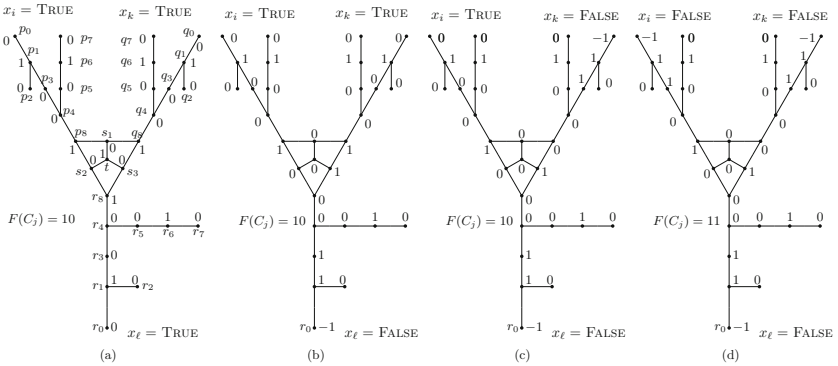


Fig. 3. Gadget C_j for clause c_j: (a) $x_i = x_k = x_\ell =$ True; (b) $x_i = x_k =$ True and $x_\ell =$ False; (c) $x_i =$ True and $x_k = x_\ell =$ False; (d) $x_i = x_k = x_\ell =$ False

to a vertex in V_ℓ which represents $x_{\ell,j}$; otherwise, if x_ℓ is False, then r_0 connects to a vertex in V_ℓ which represents $\overline{x_{\ell,j}}$. This completes the construction of the clause gadget.

We denote the minimum weight for the clause gadget by

$$F(C_j) = \min_f \{ \sum_x f(x) \mid x \in \{p_1, \ldots, p_8, q_1, \ldots, q_8, r_1, \ldots, r_8, s_1, s_2, s_3, t\} \}.$$

By Lemma 2, we assign $f(p_1) = f(q_1) = f(r_1) = 1$, $f(p_2) = f(q_2) = f(r_2) = 0$, $f(p_6) = f(q_6) = f(r_6) = 1$ and $f(p_7) = f(q_7) = f(r_7) = 0$. Then since $\sum_{x \in N[p_4]} f(x) \geq 1$, $\sum_{x \in N[q_4]} f(x) \geq 1$, $\sum_{x \in N[r_4]} f(x) \geq 1$ and $\sum_{x \in N[s_1]} f(x) \geq 1$, thus $F(C_j) \geq 10$. Here we claim that if a clause is True, then $F(C_j)$ meets the lower bound, that is, $F(C_j) = 10$; otherwise, if a clause is False, then $F(C_j) = 11$.

To prove $F(C_j) = 10$ for a True clause gadget, we show that there exists an assignment of $f(x)$ such that $\sum_{x \in N[p_4]} f(x) = 1$, $\sum_{x \in N[q_4]} f(x) = 1$, $\sum_{x \in N[r_4]} f(x) = 1$, and $\sum_{x \in N[t]} f(x) = 1$. To prove $F(C_j) = 11$ for a FALSE clause gadget, we show that there exists an assignment of $f(x)$ such that one of $\sum_{x \in N[p_4]} f(x)$, $\sum_{x \in N[q_4]} f(x)$, $\sum_{x \in N[r_4]} f(x)$, and $\sum_{x \in N[t]} f(x)$ is two, and the rest of them are one.

Analysis of Truth Assignment. We need to analyze totally four cases for the truth assignments for the clause gadget mentioned above. We discuss the case that all three literals are False in the following paragraph since it is the most important case. As for the other three cases, we omit the analysis.

Suppose that all three literals x_i, x_k and x_ℓ are False as shown in Figure 3(d). We prove that $F(C_j) = 10$ is impossible as follows. Suppose that $F(C_j) = 10$. Then $\sum_{x \in N[y]} f(x) = 1$ for $y \in \{p_4, q_4, r_4, t\}$. Since x_i, x_k and x_ℓ are False, we assign $f(x)$ for $x \in N[p_4]$ by setting $f(p_3) = 1$ and $f(p_4) = f(p_5) = 0$, and we perform the similar labeling for $x \in N[q_4]$ and for $x \in N[r_4]$. On the other hand, it is easy to check that the minimum sum of weight of vertices in $\{p_8, q_8, r_8, s_1, s_2, s_3, t\}$ is not less than 2. Hence, $F(C_j) \geq 11$.

In a true assignment of clause c_j, the minimum weight of $F(C_j)$ for the optimal minus dominating function for gadget C_j is 10, which is omitted due to lack of space. Hence, the Boolean expression Φ is satisfied if and only if the constructed graph has a minus dominating function of weight $n(8m+4)+10m$. □

4 APX-hardness for Graphs of Maximum Degree 7

In this section, we show that the MD problem for graphs of maximum degree 7 is APX-hard. Alimonti and Kann [1] show that the domination problem is APX-hard on subcubic graphs. Here we use a well-known technique called *L-reduction* to show the APX-hardness for the MD problem. See [8] for more details on L-reduction. We show that our problem satisfies the two main properties of L-reduction.

We perform an L-reduction f from an instance of the domination problem on subcubic graphs to the corresponding instance of the MD problem on graphs of maximum degree 7.

Given a subcubic graph $G = (V, E)$, we construct a graph $G' = (V', E')$ as follows. For each vertex v in V, we add $\deg_G(v) + 1$ keys as Figure 1(b) and connect a_7 of each key to v. This completes the construction of G'. For each vertex $v \in V$, we have just added one more edge connecting to the vertex v for each edge adjacent to v in graph G. Since graph G is subcubic, we have that G' is of maximum degree 7. Next, we obtain the following lemma.

Lemma 3. *Let $G = (V, E)$ be a subcubic graph and let the corresponding $G' = (V', E')$ constructed as described above. If D^* is a minimum dominating set of G, and f^* is a minimum minus dominating function of G'. Then $w(f^*(V')) = |D^*| + 6m + 2n$, where $n = |V|$ and $m = |E|$.*

Proof. We construct a function f by assigning the vertices $\{a_1, a_3, a_6, a_7\}$ in each keys with value $+1$, we label D^* and the vertices $\{a_2, a_4\}$ in each keys with value 0, and label other vertices with value -1. Then we verify whether function f is a valid minus dominating function of G', that is, we verify whether $N_G[x] > 0$ for each vertex x in G'.

It is clear that the sum of function values of the neighborhood of each vertex in keys is greater than 0. Now we claim that the same holds for the remaining vertices in V'. For a vertex v labeled with 0, the vertex has at most $\deg_G(v)$ neighbors labeled with -1, since there are $\deg_G(v) + 1$ keys connecting to v, we obtain that $N_G[x] > 0$. Moreover, we consider au vertex labeled with -1 in V'. Since the corresponding vertex of u in G is not in D^*, there must be at least a neighbor of u in G' labeled with value 0. Hence, the sum of the function values of the closed neighborhood of u is positive. Then we calculate the weight for minus dominating function f, $w(f) = 3(2m+n) - (n - |D^*|) = 6m + 2n + |D^*|$. Thus we have $w(f^*) \leq w(f) = 3(2m + n) - (n - k) = 6m + 2n + |D^*|$.

Now let f^* be the minimum minus dominating function of G', let vertex set D in G collect those vertices labeled with 0 and $+1$ in f^*. Then we claim that D is a dominating set of G, in other words, we claim that for each vertex v in

$G \setminus D$, there is at least a neighbor in D. Let v' be a vertex labeled with -1 in G'. Note that there are $2\deg_G(v)+2$ vertices in $N_{G'}[v']$. It is clear that the vertices in $N_{G'}[v'] \cap F \setminus \{v\}$ must be labeled with $+1$. Since f^* is a minus dominating function, the sum of $N_{G'}[v']$ must be positive. Hence, there must exist at least one neighbor of $v \notin F$ is labeled with 0 or $+1$. That is, D is a dominating set of G. Then we calculate the size of set D. $|D| \leq w(f^*) - 3(2m+n) + n = w(f^*) - 6m - 2n$. Thus we have $|D^*| \leq |D| \leq w(f^*) - 6m - 2n$. Hence we have $w(f^*(V')) = |D^*| + 6m + 2n$. This completes the proof. □

Since G is a subcubic graph, we have $m \leq \frac{3n}{2}$. Moreover, according to Lemma 4, $n \leq 5|D^*|$. Hence, $w(f^*) = |D^*| + 6m + 2n \leq |D^*| + 9n + 2n \leq |D^*| + 55|D^*| = 56|D^*|$. Since $w(f^*) \leq 56|D^*|$, then $\alpha = 56$.

Now we consider a minus dominating function f of G', we can construct a dominating set D for G by the above algorithm, then we have $|D| = w(f) - 6m - 2n$. Thus we obtain that $|D| - |D^*| = w(f) - 6m - 2n - (w(f^*) - 6m - 2n) = w(f) - w(f^*)$ Thus $\beta = 1$. Hence, we have proved that f is an L-reduction with $\alpha = 56$ and $\beta = 1$. Finally, we obtain the following theorem.

Theorem 3. *The MD problem is APX-hard for graphs of maximum degree* 7.

5 An FPT-algorithm for Subcubic Graphs

In Section 2, we have shown that the MD problem for subcubic bipartite planar graphs is NP-complete. It thus follows that the MD problem for subcubic graphs is NP-complete. Then it is interesting to study FPT-algorithms for the MD problem on subcubic graphs parameterized by the MD number. Th the best of our knowledge, there is no such FPT-algorithm in the literature.

In this section, we thus present the first FPT-algorithm for the MD problem on subcubic graphs G parameterized by the MD number k. In the following lemma, we begin with showing that our problem has a kernel of size $5k$. Then with an involved analysis, we come up with an FPT-algorithm which runs in time $O^*(2.3761^{5k})$.

Lemma 4. *There is a kernel of size* $5k$ *for the MD problem on subcubic graphs, and the bound is tight, where* k *is the weight for the minus dominating function of graph* G.

Proof. Let $G = (V, E)$ be a subcubic graph. For any minus dominating function of G with weight k, we claim that $|V| \leq 5k$. We divide weight k into two parts,

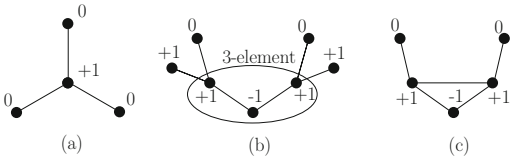


Fig. 4. (a)(b) The two conditions for weight $+1$. (c) A minus dominating function of five vertices with weight 1.

such that one part contains any vertex labeled with $+1$, whose neighbors are all not labeled with -1, and the other part contains any vertex labeled with $+1$, which has a neighbor labeled with -1. For the first part, If every single value one of the weight k is from this part, each value one contains at most four vertices see Figure 4(a), let V_1 be a set contains these vertices, we have $n \leq 4k$. Second, let v be a vertex labeled with value -1, it is clear that v has at least two neighbors labeled with value $+1$, and the distance to any other vertex labeled with -1 is at least three. Thus we obtain a 3-*element*, which contains v labeled with value -1 and its two neighbors see Figure 4(b), which labeled with $+1$. Moreover, for any pair of such subsets, their intersection is empty. Let u be the vertices labeled with $+1$, which is neighboring 3-element, it is clear that u belongs to a 3-element or to V_1. Next we consider the vertices labeled with 0, which connect to 3-element. We can observe that the vertices which labeled with 0 and connect to vertex labeled with -1 in 3-element, the vertices either the vertex connect to $+1$ in 3-element or the vertex labeled 0 in V_1. So the second part contains at most five vertices. If every single value of the weight k is from the second part, we have $n \leq 5k$. Finally, we can observe that all the vertices are concerned in the above two part. Hence, we can obtain a kernel of size $5k$ as an upper bound for the MD problem on subcubic graphs. Furthermore, For a graph with five vertices labeled as in Figure 4(c) is $k = 1$ and $n = 5$, so the bound of kernel is tight. □

According to this lemma, a naïve FPT-algorithm can be easily obtained via the brute-force method which runs in $O^*(3^{5k}) = O^*(243^k)$ time. Contrary to the brute-force algorithm, we claim that our FPT-algorithm runs in time $O^*(2.3761^{5k}) = O^*(75.7397^k)$, which is a great improvement. Now, our FPT-algorithm is presented in the following theorem. We first give the detailed algorithm and its correctness proof, and then analyze its time complexity.

Theorem 4. *The MD problem for subcubic graphs G can be solved in $O^*(2.3761^{5k})$ time, where k is the MD number of G.*

Proof. Due to Lemma 4, we only need to consider the given subcubic graph G with kernel size of $5k$. Since disconnected components of a graph can be handled separately, we assume that the given graph G is connected in the following context. We also use $N[\cdot]$ to represent $N_G[\cdot]$ for simplicity.

The details of our algorithm are as follows. In our algorithm, we grow a potential optimal minus dominating set D incrementally, where the *minus dominating set* D is a subset of vertices in V labeled with values $+1, 0$ or -1. The *label* of a vertex is the value of vertex assigned by a specific minus dominating function of G. A vertex is called *labeled* if it has been assigned a value; otherwise, it is called *unlabeled*. The weight for the closed neighborhood $N[v]$ of a labeled vertex v is called *valid* (resp. *invalid*) if all the vertices in $N[v]$ are labeled, and the sum of weights of the vertices in $N[v]$ is positive (resp. non-positive). In the process, we maintain a list L of unlabeled vertices which are the neighboring vertices of the currently labeled vertices in D. Initially, L is set to contain one degree-3 vertex of the input graph G, and $D = \emptyset$. During each iteration of our algorithm, we select

an arbitrary unlabeled vertex y from list L as the focus vertex, and we assume that x is a labeled vertex adjacent to y in $D \cap N[y]$. We set $\Delta = N[N[y]] \setminus D$. Then our algorithm makes execution branches on all different ways of assigning values to vertices in Δ of new unlabeled vertices in $N[N[y]] \setminus D$, we performing detailed case analysis from *Case 1* to *Case 15*. In the case analysis, our algorithm makes subsequent recursions only on those feasible ways of value assignment in the corresponding cases. For each of such subsequent recursions, the vertices of Δ, which have been labeled, are added into D, resulting in a larger labeled dominating set $D + \Delta$. Then for each vertex v in $D + \Delta$, if all the vertices in $N[v]$ are labeled, we then check whether the weight for $N[v]$ is valid. If we reach any weight for closed neighborhood of a vertex, which is invalid, then the current execution branch is aborted; otherwise, we proceed to update D and L for the next round of execution. We update D by setting $D = D + \Delta$, and then we update list L accordingly by visiting the neighboring vertices of the neighbors of vertices in Δ. More precisely, the vertices in Δ are removed from L, and the unlabeled vertices in the neighborhoods of vertices in Δ are added into L. It is clear that such an update takes only $O(1)$ time. We then proceed to the next execution round with the updated D and L as parameters. We repeat such a selection step until all vertices in G are labeled, that is, L becomes empty. Thus we obtain a candidate minus dominating set D.

In the selection process, we enumerate and store all possible candidates of D according to the above recursive procedure. When all branches of the selection process finished, we obtain a set of candidate minus dominating sets D for the input graph G. We choose the one with minimum weight among all these candidates. This completes our algorithm.

Case analysis for selection step. We divide the analysis into two parts: the initial step and the general selection step.

The initial step. We choose one degree-3 vertex v is placed in D. Then add one neighbor u of v is subsequently into D. In these two beginning steps, there are 8 choices to labeled vertices u and v, since u and v cannot both be of value -1. In any subsequent step, we focus on an unlabeled vertex y, which has a neighbor x in D. Now the degree of x and y can be one, two or three. However, it is easy to see that the worst case happens when both degree of x and y are three. We only need to perform the detailed case analysis for such a case. Thus in the following, we assume that the degree of both x and y are three. Due to the above initial step, we know that x must have another neighbor x_1 in D. Thus we have in total 15 cases to analyze by considering which vertices of $N[N[y]]$ lie in d. See Figure 5.

The selection step. We analyze all possible labeling ways to find the optimal minus dominating function. Due to lack of space, we only provide the analysis of *Case 1* (Figure 5(a)) in the following. The analysis of *Cases 2* to *15* are omitted.

Case 1. Let x_2 be the third neighbor of x, let y_1 and y_2 be the other two neighbors of y, and let z_1, z_2 be the neighbors of y_1, z_3, z_4 be the neighbors of y_2. Since y_1 and y_2 are symmetric, we need to consider 15 cases depending on the

content of Δ, the set of unlabeled vertices in the subgraph of G. See Figures 5(a) to (o). $\Delta = \{y, x_2, y_1, y_2, z_1, z_2, z_3, z_4\}$. That is, $\{x, x_1\} \subset D$. See Figure 5(a).

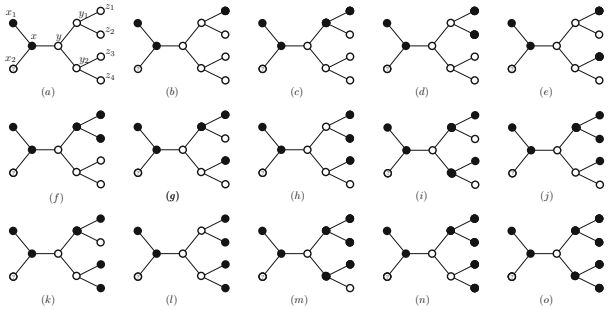


Fig. 5. This figure shows the 15 cases, where the labeled vertices in D are drawn as black disks, the unlabeled vertices in Δ as circles, and the question mark in circle is where we analyze whether the vertex is labeled or unlabeled in this case

In order to reduce the number of cases we need to consider under a specific case, we only need to consider the worst-case scenarios in each of the cases. For such a purpose, we make the following three assumptions. We remark that the three assumptions will be applied to all the subsequent cases in this proof.

(i) If there is an edge not in the subgraph, let u, v be the endpoints. Then we can add two vertices u' and v', where we connect u to u' and connect v to v'. Thus we can label the subgraph with constrain : the value of u and v' is same, so as v and u'. We can observe that the number of feasible ways of labeling the vertices with constrain is less than the vertices without constrains, which we will consider in other cases. Hence, we do not need to consider adding some edges in subgraphs.

(ii) We observe that a vertex labeled with -1 need to have at least two vertices labeled with $+1$ as its neighbors, and a vertex labeled with $+1$ can have vertices labeled with $+1, 0$ or -1 as its neighbors. Thus for a specific case, the worst-case scenario for the number of feasible ways to label the vertices in Δ is when the labeled vertices for the specific case (for instance, vertices x and y in *Case 1*) are all assigned value $+1$.

(iii) We observe that, for a graph G, if there is a connected graph H is delete a vertex v from G, the number choices to labeling vertices in H are less than or equal to number of choices of graph to labeling G with v assigned value $+1$. Hence, we do not need to consider the subgraph which delete some vertices.

For *Case 1*, according to assumption (i), we suppose that there is no edge connecting any pair of vertices in the subgraph, and according to assumption (ii), we suppose that x and x_1 are labeled with $+1$. We have two situations depending on whether x_2 is labeled. In *Case 1*(a), we first consider x_2 is unlabeled, thus x_2 has at most 3 choices for its labeling, say with value -1, 0 or $+1$. Now we focus on vertex y_1. First we consider to label y_1 with value -1, then $\{z_1, z_2\}$ has at

most 3 choices for assigning, and $\{y_2, z_3, z_4$ has at most 14 choices. Thus there are at most 42 choices if y_1 is labeled with -1. Second, we consider to label y_1 with value 0, then $\{z_1, z_2\}$ has at most 6 choices for assigning the values, and $\{y_2, z_3, z_4\}$ has at most 17 choices; hence, there are at most 102 choice if y_1 is labeled 0. Lastly, we consider to label y_1 with value $+1$, then $\{z_1, z_2\}$ has at most 8 choices for assigning the values, and $\{y_2, z_3, z_4\}$ has at most 17 choices. Thus there are at most 136 choice if y_1 is labeled $+1$. After all of $x_2, y_1, y_2, z_1, z_2, z_3, z_4$ are labeled, it is clear that we can label vertex y with an unique minimum value, such that the weights for neighborhoods of x, y, y_1 and y_2 are positive, respectively. By multiplying with the three choices for the value of x_2. Thus there are at most $3 \times (42+102+136) = 840$ feasible ways in total to label the eight vertices in Δ for *Case 1*(a).

For *Case 1*(b), where x_2 is labeled, we use similar argument as the analysis for *Case 1*(a). But we do not need to multiply three choices for the value of x_2. Thus there are at most 240 feasible ways in total to label the seven vertices in Δ for *Case 1*(b). This finishes the analysis of *Case 1*.

In the above detailed analysis, we obtain the recurrence relation $T(n) \leq 840(n-8) + O(1)$ for the worst-case running time of *Case 1*(a). By analyzing all 15 cases in Figure 5 and solving the corresponding recurrence relations, we thus have the worst-case running time for the whole algorithm, which occurs at *Case 5*(a) (see Figure 5(e)). Hence we obtain that the total running time of our algorithm is $T(n) = O^*(2.3761^n) = O^*(2.3761^{5k})$. □

References

1. Alimonti, P., Kann, V.: Hardness of approximating problems on cubic graphs. In: Proc. 3rd Italian Conference on Algorithms and Complexity (CIAC), pp. 288–298 (1997)
2. Damaschke, P.: Minus domination in small-degree graphs. Discrete Applied Mathematics **108**(1–2), 53–64 (2001)
3. Downey, R.G., Fellows, M.R.: Parameterized Complexity, 3rd edn. Springer (1999)
4. Dunbar, J., Goddard, W., Hedetniemi, S., McRae, A., Henning, M.A.: The algorithmic complexity of minus domination in graphs. Discrete Applied Mathematics **68**(1–2), 73–84 (1996)
5. Dunbar, J., Hedetniemi, S., McRae, A., Henning, M.A.: Minus domination in graphs. Discrete Applied Mathematics **199**(1–3), 35–47 (1999)
6. Faria, L., Hon, W.-K., Kloks, T., Liu, H.-H., Wang, T.-M., Wang, Y.-L.: On complexities of minus domination. In: Widmayer, P., Xu, Y., Zhu, B. (eds.) COCOA 2013. LNCS, vol. 8287, pp. 178–189. Springer, Heidelberg (2013)
7. Lichtenstein, D.: Planar formulae and their uses. SIAM Journal on Computing **11**(2), 329–343 (1982)
8. Papadimitriou, C.H., Yannakakis, M.: Optimization, approximation, and complexity classes. Journal of Computer and System Sciences **43**(3), 425–440 (1991)
9. Zheng, Y., Wang, J., Feng, Q., Chen, J.: FPT results for signed domination. In: Agrawal, M., Cooper, S.B., Li, A. (eds.) TAMC 2012. LNCS, vol. 7287, pp. 572–583. Springer, Heidelberg (2012)