

Formalization of the generalized domination structure in graphs with combinatorial analysis

Misa Nakanishi *

Abstract

Following [3], we provide a technical paper. This paper addresses the minimum dominating set problem and generalizes the results for graphs with maximum degree 3 to general graphs. In addition, it complements the technical aspects of the results.

keywords : minimum dominating set, structure, NP-complete

MSC : 05C69

1 Introduction

This technical paper addresses the minimum dominating set problem and aims to determine the minimum dominating set of a graph, complementing and extending the results of [3]. A dominating set of a graph G is a set S of vertices of G such that every vertex v of G is either in S or adjacent to a vertex of S . A minimum dominating set is a dominating set of minimum size.

The minimum dominating set problem for cubic graphs is known to be NP-complete [1], which implies that the problem is NP-complete for general graphs as well. In this work, we extend the degree-decreasing sequence used in the proof of the theorem in [3], thereby generalizing the result from graphs of maximum degree 3 to arbitrary graphs.

As a main part, this paper introduces three polynomial-time algorithms for graphs and provides technical refinements, ultimately leading that the problem lies in P, thus establishing that $P = NP$.

2 Notation

In this paper, a graph G is finite, undirected, and simple with the vertex set V and edge set E . We follow the notations presented in [2]. For a vertex $v \in V(G)$, the open neighborhood, denoted by $N_G(v)$, is $\{u \in V(G) : uv \in E(G)\}$, and the closed neighborhood, denoted by $N_G[v]$, is $N_G(v) \cup \{v\}$, also for a set $W \subseteq V(G)$, let $N_G(W) = \bigcup_{v \in W} N_G(v)$ and $N_G[W] = N_G(W) \cup W$. A *dominating set* $X \subseteq V(G)$ is such that $N_G[X] = V(G)$. A minimum dominating set, called

*E-mail address : nakanishi@mx-keio.net

a *d-set*, is a dominating set of minimum size. A *dominating vertex* is a vertex in the dominating set. A vertex is *dominated* by its adjacent vertex in the dominating set. *X-3-path* is a path that has *labels* on the vertices so that the distance between the labels is 3 if the length of the path is at least 2, otherwise *X-3-path* is a path of length 1 that has at most one label on the vertices. Two cycles C_1 and C_2 are said to be *connecting without seams* if *X-3-paths* can be assigned to C_1 and C_2 as $C_1 \cap C_2$ is one *X-3-path*.

3 Algorithms and complements

We consider a connected graph G , otherwise consider each component one by one. First, we introduce the algorithm Scheme **K** as follows.

Algorithm 1: Scheme **K**: construction applied to a connected graph G - Part 1

Input : A connected graph G
Output: A resulting graph $\mathbf{K}(G)$ and a tree $T(G)$

- 1 **if** $|V(G)| \leq 2$ **then**
- 2 $T \leftarrow \emptyset$;
- 3 assign a label to $v \in V(G)$;
- 4 add a vertex to T ;
- 5 **return** G as $\mathbf{K}(G)$ and T as $T(G)$;
- 6 $G_0 \leftarrow G$ and $k \leftarrow 0$;
- 7 $T_0 \leftarrow \emptyset$ and $l \leftarrow 0$;
- 8 **repeat**
- 9 Let x be a cut vertex of G_k .
- 10 **foreach** pairs of distinct components C_1, C_2 of $G_k - x$ **do**
- 11 **foreach** pairs of vertices $u \in V(C_1), v \in V(C_2)$ **do**
- 12 add an edge uv to G_k ;
- 13 $k \leftarrow k + 1$;
- 14 **until** there exists a cut vertex in G_k
- 15 Find an induced cycle C in G_k .
- 16 **if** length of C is $0 \pmod 3$ **then**
- 17 assign X -3-path to C in G_k ;
- 18 add C as a vertex to T_l ;
- 19 $k \leftarrow k + 1$ and $l \leftarrow l + 1$;
- 20 **else if** length of C is $2 \pmod 3$ **then**
- 21 take a 4-path $x_1x_2x_3x_4 \subseteq C$;
- 22 add edges x_1x_3, x_2x_4, x_1x_4 to G_k ;
- 23 assign X -3-paths to $0 \pmod 3$ induced cycles connecting without seams in the formed subgraph (call this subgraph (a));
- 24 add each $0 \pmod 3$ induced cycle as a vertex and one of its connections with the $0 \pmod 3$ induced cycles in the subgraph (a) as an edge to T_l ;
- 25 $k \leftarrow k + 1$ and $l \leftarrow l + 1$;
- 26 **else**
- 27 take a 4-path $x_1x_2x_3x_4 \subseteq C$;
- 28 add edges x_1x_3 and x_2x_4 to G_k ;
- 29 assign X -3-paths to $0 \pmod 3$ induced cycles connecting without seams in the formed subgraph (call this subgraph (b));
- 30 add each $0 \pmod 3$ induced cycle as a vertex and one of its connections with the $0 \pmod 3$ induced cycles in the subgraph (b) as an edge to T_l ;
- 31 $k \leftarrow k + 1$ and $l \leftarrow l + 1$;
- 32 Find a shortest path P over all two vertices v_1 and v_2 lying on previously assigned X -3-path in G_k such that X -3-path has not been assigned to $\dot{v}_1P\dot{v}_2$ yet.

Algorithm 2: Scheme **K** - Part 2

```
1 if such a path  $P$  exists then
2   Let  $P'$  be the previously assigned  $X$ -3-path of length at least 2
   containing  $v_1$  and  $v_2$ .
3   if  $v_1Pv_2P'v_1$  forms a cycle of length 0 mod 3 in  $G_k$  then
4     assign  $X$ -3-path to  $v_1Pv_2P'v_1$  in  $G_k$  while maintaining or
     deleting the labels of existing  $X$ -3-path;
5     add  $v_1Pv_2P'v_1$  as a vertex and one of its connections with the 0
     mod 3 cycles which have a part of  $v_1P'v_2$  and contained as the
     vertices in  $T_l$  already as an edge to  $T_l$ ;
6      $k \leftarrow k + 1$  and  $l \leftarrow l + 1$ ;
7   else if  $v_1Pv_2P'v_1$  forms a cycle of length 2 mod 3 in  $G_k$  then
8     take a 4-path in  $v_1Pv_2P'v_1$  and add three edges to  $G_k$  as in
     subgraph (a);
9     if possible then
10      assign  $X$ -3-paths to 0 mod 3 induced cycles connecting
      without seams in the subgraph (a) while maintaining or
      deleting the labels of existing  $X$ -3-path;
11    else
12      redo step 8;
13    add each 0 mod 3 induced cycle as a vertex and one of its
     connections with the 0 mod 3 induced cycles in the subgraph
     (a) (or one of its connections with the 0 mod 3 cycles which
     have a part of  $v_1P'v_2$  and contained as the vertices in  $T_l$ 
     already) as an edge to  $T_l$ ;
14     $k \leftarrow k + 1$  and  $l \leftarrow l + 1$ ;
15   else if  $v_1Pv_2P'v_1$  forms a cycle of length 1 mod 3 in  $G_k$  then
16     take a 4-path in  $v_1Pv_2P'v_1$  and add two edges to  $G_k$  as in
     subgraph (b);
17     if possible then
18      assign  $X$ -3-paths to 0 mod 3 induced cycles connecting
      without seams in the subgraph (b) while maintaining or
      deleting the labels of existing  $X$ -3-path;
19     else
20      redo step 16;
21     add each 0 mod 3 induced cycle as a vertex and one of its
     connections with the 0 mod 3 induced cycles in the subgraph
     (b) (or one of its connections with the 0 mod 3 cycles which
     have a part of  $v_1P'v_2$  and contained as the vertices in  $T_l$ 
     already) as an edge to  $T_l$ ;
22      $k \leftarrow k + 1$  and  $l \leftarrow l + 1$ ;
23   goto step 32 of Part 1 (continue the loop);
24 else
25   break the loop;
26 return  $G_k$  as  $\mathbf{K}(G)$  and  $T_l$  as  $T(G)$ ;
```

Proposition 3.1 ([2]). *A graph is 2-connected if and only if it can be constructed from a cycle by successively adding H -paths to graphs H already constructed.*

Fact 3.1. *In a 2-connected graph, any two vertices are connected by a path along edges in some ear of its ear decomposition.*

Proof. The statement is proved by induction on ears. As a base case, if v and w in first ear (cycle), path exists trivially. Assume path exists for first k ears, add $(k + 1)$ -th ear if needed. 2-connectedness ensures every vertex is in some ear. We can concatenate paths along endpoints of ears. \square

Proposition 3.2. *In the algorithm Scheme **K** - Part 1 step 32, it is possible to find previously assigned X -3-path between v_1 and v_2 .*

Proof. By Fact 3.1, the claim holds. \square

Proposition 3.3. *In the algorithm Scheme **K** - Part 1 step 32, create a cycle by choosing the shortest path connecting the two endpoints of X -3-path. If the length of the cycle is $1 \pmod 3$ or $2 \pmod 3$, choose a 4-path and add edges according to the rule. For some 4-path in the cycle, X -3-paths can be assigned to $0 \pmod 3$ induced cycles connecting without seams in the subgraph while maintaining or deleting the labels of existing X -3-paths.*

Proof. Given the length of the existing X -3-path X ($0, 1, 2 \pmod 3$), the labeling method Y (3 rotation), the length of the cycle Z ($1, 2 \pmod 3$), and the 4-path selection method A ($\mathcal{O}(n^4)$ for $n = |V(G)|$), we obtain the proposition through simple case analysis of $X \times Y \times Z \times A$. \square

Note that $\mathbf{K}(G)$ and $T(G)$ are not unique and constructed from G arbitrarily. By the scheme, $\mathbf{K}(G)$ is 2-connected with $V(G) = V(\mathbf{K}(G))$ and $E(G) \subseteq E(\mathbf{K}(G))$, and all edges in $\mathbf{K}(G)$ are covered by at least one set of X -3-paths but may be covered by other set of X -3-paths with the rotation of labels. We have a tree-structured list $T(G)$ of $0 \pmod 3$ cycles connecting without seams. The label rotation follows this list. Choose an initial vertex to label in $V(G)$, and from there, assign X -3-path to the cycle in the list.

Consider labeling vertices by assigning X -3-paths to the cycles of length $0 \pmod 3$ connecting without seams in $\mathbf{K}(G)$ following $T(G)$. Note that certain cycles may not be counted for the labeling, and have as few labeled vertices as possible.



Figure 1. Labeling vertices by the procedure for labeling (*).

Remark 3.1 ([3]). *Let X be a dominating set of G . Every subset $D \subseteq X$ is a d -set of $G[N_G[D]]$ if and only if X is a d -set of G .*

Proposition 3.4. (i) *For every labeling, the set of all labeled vertices is a minimal dominating set of $\mathbf{K}(G)$. In addition, by selecting the first vertex to label, the remaining labeled vertices are uniquely determined.* (ii) *For at least one labeling, the set of all labeled vertices is a d -set of $\mathbf{K}(G)$.*

Proof. Since every edge in $\mathbf{K}(G)$ is covered by X -3-path, for every labeling, the set of all labeled vertices is an independent dominating set of $\mathbf{K}(G)$, that is, a minimal dominating set of $\mathbf{K}(G)$. After selecting the vertex to label first, if there are multiple ways to label vertices during the labeling process, those vertices are two adjacent vertices in $K^4 - e \subseteq \mathbf{K}(G)$ for some $e \in E(K^4)$. Now, we take the procedure for labeling (*): Let U_0 be the set of all unlabeled vertices not dominated by the labeled vertices. For the two vertices y_1 and y_2 , select y_1 if $N_{\mathbf{K}(G)}[y_2] \cap U_0 \subseteq N_{\mathbf{K}(G)}[y_1]$, select y_2 if $N_{\mathbf{K}(G)}[y_1] \cap U_0 \subseteq N_{\mathbf{K}(G)}[y_2]$, select the first taken vertex y_1 or y_2 otherwise. Then the remaining labeled vertices are determined following $T(G)$ uniquely. Therefore, the statement (i) holds. By Remark 3.1, the statement (ii) obviously holds. \square

Let Y be a d -set of $\mathbf{K}(G)$ that is obtained by labeling. Let \mathcal{Y} be the set of all Y . Suppose G has a cut vertex v and $w \in N_G(v)$ is not a cut vertex. For $Y_1, Y_2 \in \mathcal{Y}$ such that $Y_1 \cap Y_2 \neq \emptyset$, if $v \in Y_1 \setminus Y_2$ and $w \in Y_2 \setminus Y_1$, then let \mathcal{Y} be $\mathcal{Y} \setminus Y_2$. Repeat this operation while they occur and let $\mathcal{A} = \mathcal{Y}$.

Proposition 3.5. *Suppose G has a cut vertex v and $w \in N_G(v)$ is not a cut vertex. For $Y_1, Y_2 \in \mathcal{Y}$ such that $Y_1 \cap Y_2 \neq \emptyset$, if $v \in Y_1 \setminus Y_2$ and $w \in Y_2 \setminus Y_1$, then Y_1 is better for a d -set of G .*

Proof. In the 0 mod 3 induced cycles constructed from v by the edge addition rules of the algorithm Scheme \mathbf{K} - Part 1 step 8, w does not dominate any other neighbor of v in G . \square

Definition 3.1. *For $\mathcal{A}' \subseteq \mathcal{A}$, define \equiv so that two sets $A_1, A_2 \in \mathcal{A}'$ are equivalent if and only if every pair of sets in \mathcal{A}' has nonempty intersection.*

Proposition 3.6. *Let X be a d -set of G . Let \mathcal{X} be the set of all X . For some $\mathcal{B} \in \mathcal{A}/\equiv$ and for each $Y \in \mathcal{B}$, there exists $X \in \mathcal{X}$ such that $Y = X$, otherwise, for each $Y \in \mathcal{A}$, there exists $X \in \mathcal{X}$ such that $Y \subseteq X$.*

Proof. Let $\mathcal{B} \in \mathcal{A}/\equiv$. Consider labeling vertices in $\mathbf{K}(G)$. For any $Y_1, Y_2 \in \mathcal{B}$ such that $Y_1 \neq Y_2$, the vertices of $Y_1 \setminus Y_2$ and $Y_2 \setminus Y_1$ are interchangeable and equivalent. By the definition of $\mathbf{K}(G)$ and \mathcal{B} , it suffices that considering subgraph (a) or (b) of $\mathbf{K}(G)$. Now, $N_{\mathbf{K}(G)}[Y_1 \setminus Y_2] = N_{\mathbf{K}(G)}[Y_2 \setminus Y_1]$. If Y_1 is a dominating set of G , then for some $X \in \mathcal{X}$, $X = Y_1$. Now, $N_G[Y_2 \setminus Y_1] \setminus N_G[Y_1 \setminus Y_2] \subseteq N_G[Y_1] = V(G)$. By considering the subgraph (a), we have $N_G[Y_1 \setminus Y_2] \setminus N_G[Y_2 \setminus Y_1] \subseteq N_G[Y_2]$. Hence, $N_G[Y_1] = N_G[Y_2]$. Suppose that for all $Y \in \mathcal{A}$, Y is not a dominating set of G . Now, each Y_1 and Y_2 is not a dominating set of G . By considering the subgraph (b), we have $N_G[Y_2 \setminus Y_1] \setminus N_G[Y_1 \setminus Y_2] \not\subseteq$

$N_G[Y_1]$ and $N_G[Y_1 \setminus Y_2] \setminus N_G[Y_2 \setminus Y_1] \not\subseteq N_G[Y_2]$. Hence, $N_G[Y_1] \setminus N_G[Y_2] \neq \emptyset$ and $N_G[Y_2] \setminus N_G[Y_1] \neq \emptyset$. Let $\mathcal{B}_1, \mathcal{B}_2 \in \mathcal{A}/\equiv$ with $\mathcal{B}_1 \neq \mathcal{B}_2$. For any $Y_3 \in \mathcal{B}_1$ and any $Y_4 \in \mathcal{B}_2$, since $Y_3 \cap Y_4 = \emptyset$ and each Y_3 and Y_4 is not a dominating set of G , it follows that $N_G[Y_3] \setminus N_G[Y_4] \neq \emptyset$ and $N_G[Y_4] \setminus N_G[Y_3] \neq \emptyset$. Now, Y is a d-set of $G[N_G[Y]]$. Let W be a subset of $V(G) \setminus Y$ with minimum size such that $Y \cup W$ is a dominating set of G . By Remark 3.1, $Y \cup W$ is a d-set of G . \square

Proposition 3.7. *Every $\mathcal{B} \in \mathcal{A}/\equiv$ is determined in polynomial time and $|\mathcal{A}/\equiv| \leq |V(G)|$.*

Proof. By the algorithm Scheme **K**, $\mathbf{K}(G)$ and $T(G)$ are constructed from G in polynomial time. By Proposition 3.4, for each $v \in V(G)$ to label first, the equivalence class $\mathcal{B} \in \mathcal{A}/\equiv$ is defined as for the label set $B(v)$, $\{B(v)\} = \mathcal{B}$, if $B(v)$ is a d-set of $\mathbf{K}(G)$ and $B(v) \in \mathcal{A}$. Therefore, $\{B(v)\} = \mathcal{B} \in \mathcal{A}/\equiv$ is determined in polynomial time and $|\mathcal{A}/\equiv| \leq |V(G)|$. \square

Second, we introduce the algorithm Labeling-and-Prune procedure as follows. It returns a label set $L(G)$ of G , which is $Y \in \mathcal{B} \in \mathcal{A}/\equiv$ contained as the subset in a d-set of G and is a d-set of $\mathbf{K}(G)$.

Algorithm 3: Labeling-and-Prune procedure: labeling method on $\mathbf{K}(G)$

Input : A connected graph G , $\mathbf{K}(G)$, $T(G)$
Output: A label set $L(G)$

- 1 **if** $|V(G)| \leq 2$ **then**
- 2 $L \leftarrow \emptyset$;
- 3 add a vertex in $V(G)$ to L ;
- 4 **return** L as $L(G)$;
- 5 $\mathcal{L} \leftarrow \emptyset$;
- 6 **foreach** $v \in V(G)$ **do**
- 7 choose v as the initially labelled vertex;
- 8 create a temporary labelling L ;
- 9 assign X -3-path to an $0 \bmod 3$ cycle C contained as the vertex in $T(G)$ such that $v \in C$ and add labels to L ;
- 10 **for** each $0 \bmod 3$ cycle D contained as the adjacent vertex in $T(G)$ **do**
- 11 assign X -3-path to D while maintaining or deleting the labels of existing X -3-path, following the procedure for labeling (*);
- 12 add labels to L ;
- 13 add L to \mathcal{L} and record its size $|L|$;
- 14 $\mathcal{Y} \leftarrow \{Y \in \mathcal{L}: |Y| \text{ is minimum}\}$;
- 15 **repeat**
- 16 **foreach** cut vertex v of G **do**
- 17 **foreach** neighbor $w \in N_G(v)$ with w not a cut vertex **do**
- 18 **foreach** $Y_1, Y_2 \in \mathcal{Y}$ with $Y_1 \cap Y_2 \neq \emptyset$ **do**
- 19 **if** $v \in Y_1 \setminus Y_2$ **and** $w \in Y_2 \setminus Y_1$ **then**
- 20 $\mathcal{Y} \leftarrow \mathcal{Y} \setminus \{Y_2\}$;
- 21 **until** no change to \mathcal{Y}
- 22 $\mathcal{A} \leftarrow \mathcal{Y}$;
- 23 **foreach** $Y \in \mathcal{A}$ **do**
- 24 **if** Y is a dominating set of G **then**
- 25 **return** Y as $L(G)$;
- 26 choose any $Y \in \mathcal{A}$ and **return** Y as $L(G)$;

Proposition 3.8. *In the algorithm Labeling-and-Prune procedure, $L(G)$ is computed in polynomial time.*

Proof. By Proposition 3.4, $|\mathcal{L}| \leq |V(G)|$. The combinations of a cut vertex and its neighbor are $\mathcal{O}(n^2)$ for $n = |V(G)|$. By Proposition 3.6 and Proposition 3.7, $|\mathcal{A}| \leq |V(G)|$, so the claim holds. \square

Subsequently, we determine the remaining vertices of a d -set of G that are not contained in $L(G)$ and prepare two propositions.

Proposition 3.9 ([3]). *Let $Y = L(G)$. Let G' be a graph obtained by deleting Y from, and adding edges to all vertex pairs of $\bigcup_{y \in Y} N_G(y)$ to G . Let Z_1 be a d -set of G' . Let $G'' = G - N_G[Y]$ and Z_2 be a d -set of G'' . If $|Z_1| < |Z_2|$, then $Y \cup Z_1$ is a d -set of G , and $Z_1 \cap \bigcup_{y \in Y} N_G(y) \neq \emptyset$. If $|Z_1| \geq |Z_2|$, then $Y \cup Z_2$ is a d -set of G .*

Fact 3.2. *A vertex in G is either a dominating vertex, dominated by the endpoints of some edge in G , or dominated by the endpoint of some added edge in $\mathbf{K}(G)$.*

Fact 3.3. *If a vertex of degree at least 2 and all its neighbors in G are dominated only by the endpoints of added edges in $\mathbf{K}(G)$, then that vertex and two adjacent vertices in G are three vertices in a 4-path which has added edges in the selected cycle of the algorithm Scheme \mathbf{K} .*

Proposition 3.10. *Either a vertex of maximum degree or one of its adjacent vertices is a dominating vertex or dominated by the endpoint of some edge in G .*

Proof. Let G be a connected graph. If $|V(G)| \leq 2$ then the claim obviously holds. Suppose $|V(G)| \geq 3$ and the claim does not hold. Consider added edges in $\mathbf{K}(G)$. Let v be a vertex with maximum degree d . Suppose that no vertex in $N_G[v]$ is dominated in G . An added edge connects v and its neighbors for each and dominates by its endpoint. v and its two adjacent vertices are three vertices in a 4-path which has added edges in the selected cycle C_1 for $\mathbf{K}(G)$ (according to Fact 3.3). The remaining vertex of the 4-path is either connected by some edge in G or by some added edge in $\mathbf{K}(G)$. Since one vertex of the 4-path is a dominating vertex, the claim holds if it is connected by some edge in G .

Suppose that it is connected by the added edge e . C_1 is of length $2 \pmod 3$, and the endpoint of e , say w , is also the endpoint of the added edge connected to v and is a dominating vertex.

A vertex of distance 2 from v on G , say w' , is not a dominating vertex, but is dominated by its adjacent vertex in G or by the endpoint of the added edge. In the case of the added edge, w' is connected to some edge in G as a vertex of a 4-path, a contradiction (due to Fact 3.3). w' is dominated by the endpoint of some edge in G . e is on a 4-path in another selected cycle C_2 and is an edge of triangle in $\mathbf{K}(G)$. C_2 is of length $1 \pmod 3$. w and all three neighbors and the edges between them in G are consumed by these two cycles. Now, $d \geq 3$.

Another neighbor of v on G , say x , is dominated by the endpoint y of the added edge. The added edge between v and w and the added edge between x and y are on a selected cycle C_3 via the edge vx in G . w can have at most d neighbor in G . Remaining $d - 3$ neighbor of w , if $d = 3$ then $y = w$, a contradiction. If $d > 3$, by induction, last neighbor x of v in G lead $y = w$, a contradiction. x is dominated by the endpoint of some edge in G . \square

Third, we introduce the algorithm `extendWithProp3.9` as follows. For a graph F , it returns a minimum dominating set of F , which is written by $X(F)$.

Algorithm 4: extendWithProp3.9: determination of the remaining vertices for a d-set

Input : A graph F , $L(G)$ for a component G of F
Output: A minimum dominating set $X(F)$

- 1 **if** $F = \emptyset$ **then**
- 2 \lfloor **return** \emptyset as $X(F)$;
- 3 $W \leftarrow \emptyset$;
- 4 **foreach** component G of F **do**
- 5 $S \leftarrow L(G)$;
- 6 **if** $V(G) \setminus N_G[S] = \emptyset$ **then**
- 7 $W \leftarrow W \cup S$;
- 8 **goto** step 4;
- 9 Let G' be constructed by deleting S , and for every pair $w_1, w_2 \in \bigcup_{s \in S} N_G(s)$, adding an edge $w_1 w_2$ to G .
- 10 $Z_1 \leftarrow \text{extendWithProp3.9}(G')$;
- 11 $G'' \leftarrow G - N_G[S]$;
- 12 $Z_2 \leftarrow \emptyset$;
- 13 **foreach** component C of G'' **do**
- 14 **if** $|V(C)| = 1$ **then**
- 15 $Z_2 \leftarrow Z_2 \cup V(C)$;
- 16 **else if** C is a cycle **then**
- 17 set Z to a d-set of C ;
- 18 $Z_2 \leftarrow Z_2 \cup Z$;
- 19 **else if** C is a path **then**
- 20 set Z to a d-set of C ;
- 21 $Z_2 \leftarrow Z_2 \cup Z$;
- 22 **else**
- 23 $Z \leftarrow \text{extendWithProp3.9}(C)$;
- 24 $Z_2 \leftarrow Z_2 \cup Z$;
- 25 **if** $|Z_1| < |Z_2|$ **then**
- 26 $Z_{\text{chosen}} \leftarrow Z_1$;
- 27 **else**
- 28 $Z_{\text{chosen}} \leftarrow Z_2$;
- 29 $W \leftarrow W \cup S \cup Z_{\text{chosen}}$;
- 30 **return** W as $X(F)$;

Proposition 3.11. *In the algorithm extendWithProp3.9 step 11, the maximum degree of G'' is at least one less than the maximum degree of G .*

Proof. By Proposition 3.9, it suffices that for a label set $L(G')$ of G' , $L(G') \cap \bigcup_{y \in L(G)} N_G(y) \neq \emptyset$. By Proposition 3.10, the claim holds. \square

Theorem 3.1. *The d-set of a graph is determined in polynomial time.*

Proof. It follows the algorithm `extendWithProp3.9`. The computation of $L(G)$ for a component G of the graph F is in polynomial time by Proposition 3.8. The remaining vertices of a d-set of F from $\bigcup L(G)$ are derived from Lemma 3.9 with the recursion of `extendWithProp3.9` for G' and G'' , which have the degree-decreasing sequence by Proposition 3.11. The number of recursions in `extendWithProp3.9` is $\mathcal{O}(n \log n)$ for $n = |V(G)|$. Hence, all computations in `extendWithProp3.9`, including for the number of components and the definitions of G' and G'' , can be performed in polynomial time. \square

References

- [1] P. Alimonti and V. Kann: Some APX-completeness results for cubic graphs, *Theoretical Computer Science*, 237 (2000), pp. 123-134.
- [2] R. Diestel: *Graph Theory Fourth Edition*. Springer (2010)
- [3] M. Nakanishi: Generalized domination structure in cubic graphs, *Journal of Combinatorial Mathematics and Combinatorial Computing*, 124 (2025), pp. 731-736.