$\mathcal{P}$-mixed domination: a unified approach to mixed domination and various domination-related problems in trees[†]

Chuan-Min Lee
Department of Computer and Communication Engineering
Ming Chuan University
5 De Ming Rd., Guishan District, Taoyuan County 333, Taiwan.
Tel: +886-3-350-7001 ext.3432
Fax: +886-3-359-3876
Email: joneslee@mail.mcu.edu.tw

Abstract

The literature is extensive on algorithms and complexity results for the problems of *dominating* or *covering* vertices or edges by other vertices or edges. For any one of these dominating or covering problems, we observe that no matter what the definition or concept is, it is almost always solvable in linear time for trees. In this paper, we attempt to bring together a number of recent ideas in the study of domination-related problems into a single framework for trees. To do so, we introduce the notion of *$\mathcal{P}$-mixed domination* and present a linear-time algorithm to solve the $\mathcal{P}$-mixed domination problem in trees. Our algorithm gives a unified approach to *mixed domination* and various dominating and covering problems for trees.

Keywords: Algorithm; Domination; Mixed domination; Strong elimination ordering; Tree;

1 Introduction

All graphs in this paper are simple, i.e., finite, undirected, and without self-loops or multiple edges. Let $G = (V, E)$ be a graph with vertex set V and edge set E. Unless stated otherwise, it is understood that $|V| = n$ and $|E| = m$. The vertex and edge sets of G are also referred to as $V(G)$ and $E(G)$, respectively. For an element $x \in V \cup E$, it is either a vertex or an edge in G. For simplicity, we also call it an *element* in G. Two distinct vertices u and v of G are *adjacent* if (u, v) is an edge of G. Two distinct edges e_1 and e_2 of G are *adjacent* if they have an endvertex in common. If a vertex v is connected to another vertex by an edge e, then we say that v is *incident* to e. Likewise, we say that the edge e is incident to v.

For any vertex v of a graph $G = (V, E)$, the *open neighborhood* of v in G is $N_G(v) = \{u \in V | (u, v) \in E\}$ and the *closed neighborhood* of v in G is $N_G[v] = N_G(v) \cup \{v\}$. The *degree* of a vertex v in G is $deg_G(v) = |N_G(v)|$. For an element $x \in V \cup E$, the *mixed neighborhood* of x in G is $N_G^m(x) = \{y \in V \cup E \mid y \text{ is adjacent or incident to } x\}$. The *mixed closed neighborhood* of x in G is $N_G^m[x] = N_G^m(x) \cup \{x\}$. For any vertex $v \in V$, we use $E_G(v)$ to denote the set of edges incident to v. Given two vertices u and v of G, the *distance* between them, denoted by $d_G(u, v)$, is defined as the number of edges in a shortest path between them in G. If $W \subseteq V$, then $G[W]$ denotes the subgraph of G induced by W.

A *mixed dominating set* of a graph $G = (V, E)$ is a subset D of $V \cup E$ such that $|N_G^m[x] \cap D| \geq 1$ for every element $x \in V \cup E$. The *mixed domination number* of G, denoted by $\gamma_m(G)$, is the minimum cardinality of a mixed dominating set of G. The *mixed domination problem* is to find a mixed dominating set of G of minimum cardinality. It has a practical application of placing phase measurement units to monitor the states of an electric power system [20], and has been studied or discussed in papers and books [1, 2, 4, 5, 6, 7, 9, 10, 11, 12, 14, 15, 18, 20].

Apart from the mixed domination problem, the literature is extensive on algorithms and complexity results for the problems of *dominating* or *covering* vertices or edges by other vertices or edges (cf.

[†]This research was partially supported under Research Grants NSC-101-2221-E-130-013 and NSC-102-221-E-130-004.

[5, 10, 11]). For any one of these domination or covering problems, we observe that no matter what the definition or concept is, it is almost always solvable in linear time for trees. Motivated by the observation and the recent works on mixed domination in trees [12, 20], we attempt to develop a unified approach to mixed domination and various well-known domination and covering problems for trees. To do so, we give an alternative way to define a mixed dominating set and introduce the notion of $\mathcal{P}$-*mixed domination* as below.

Mixed dominating sets can be expressed in terms of *functions*. Suppose that $G = (V, E)$ is a graph and Y is a subset of real numbers. Let $f : V \cup E \to Y$ be a function of G which assigns to each $x \in V \cup E$ a value in Y. The set Y is called the *weight set* of f. We let $f(S) = \sum_{u \in S} f(u)$ for any subset S of $V \cup E$ and let $w(f) = f(V \cup E)$ be the *weight* of f. A mixed dominating set can be viewed as a function $f : V \cup E \to \{0, 1\}$ such that $f(N_G^m[x]) \geq 1$ for every $x \in V \cup E$. The function f is called a *mixed dominating function* and $\gamma_m(G) = min\{w(f) \mid f$ is a mixed dominating function of $G\}$.

Definition 1. Let $G = (V, E)$ be a simple graph. Let I_1 and F_r be fixed integers, and let ℓ and d be fixed positive integers such that $F_r = I_1 + \ell \cdot d$. Suppose that Y is the set $\{I_1, I_1 + d, I_1 + 2d, \ldots, I_1 + (\ell - 1) \cdot d\}$ and $\mathcal{P}$ is a labeling function assigning to each element $x \in V \cup E$ a label $\mathcal{P}(x) = (t(x), k(x))$, where $t(x) \in Y \cup \{F_r\}$ and $k(x)$ is a fixed integer. For an element x in G, it is called a *free element* if $t(x) = F_r$. Otherwise, it is called a *restrictive element*. A $\mathcal{P}$-*mixed dominating function* of G is a function $f : V \cup E \to Y$ satisfying the following two conditions: (1) $f(x) = t(x)$ if $t(x) \neq F_r$, and (2) $f(N_G^m[x]) \geq k(x)$ for every element $x \in V \cup E$. The $\mathcal{P}$-*mixed domination number* of G, denoted by $\gamma_m^{\mathcal{P}}(G)$, is the minimum weight of a $\mathcal{P}$-mixed dominating function of G. The $\mathcal{P}$-*mixed domination problem* is to find a $\mathcal{P}$-mixed dominating function of G of minimum weight.

In this paper, we present a linear-time algorithm to solve the $\mathcal{P}$-mixed domination problem in trees. According to the definition, the $\mathcal{P}$-mixed domination includes the domination, k-tuple domination, $\{k\}$-domination, signed domination, minus domination, edge domination, vertex cover, and edge cover problems as special cases.

In a generic tree, we can fix a vertex of the tree to be the *root* and turn the tree into a rooted one. Clearly, this can be done in $O(n)$ time. We may therefore consider only the rooted trees in this paper.

2 An $O(n^2)$-time algorithm for $\mathcal{P}$-mixed domination in rooted trees

Let $G = (V, E)$ be a graph. A *clique* is a subset of pairwise adjacent vertices of V. A vertex v is *simplicial* if all vertices of $N_G[v]$ form a clique. The ordering $v_1, v_2, \ldots, v_n$ of the vertices of V is a *perfect elimination ordering* of G if for all $i \in \{1, \ldots, n\}$, v_i is a simplicial vertex of the subgraph G_i of G induced by $\{v_i, v_{i+1}, \ldots, v_n\}$. For each vertex $v \in \{v_i, v_{i+1}, \ldots, v_n\}$, let $N_i[v]$ denote the closed neighborhood of v in G_i. Rose [17] showed the characterization that a graph is *chordal* if and only if it has a perfect elimination ordering. A perfect elimination ordering is called a *strong elimination ordering* if it has the following property:

For each $i \leq j \leq k$ if $v_j, v_k \in N_i[v_i]$ in G_i, then $N_i[v_j] \subseteq N_i[v_k]$.

Farber [8] showed that a graph is *strongly chordal* if and only if it admits a strong elimination ordering. Currently, the fastest algorithm takes $O(m \log n)$ time [16] or $O(n^2)$ time [19] to recognize a strongly chordal graph and give a strong elimination ordering.

The *total graph* of a graph G, denoted by $T(G)$, has $V(G) \cup E(G)$ as its vertex set, and two vertices of $T(G)$ are adjacent if and only if they are adjacent or incident to each other in G. Then each element x in a graph G is the vertex x in $T(G)$, and the mixed closed neighborhood of x in G is equivalent to the closed neighborhood of x in $T(G)$.

It can be shown that the total graph of a tree is a strongly chordal graph by using the Farber's forbidden subgraph characterization of strongly chordal graphs [8].

Let $\mathcal{H} = (V_{\mathcal{H}}, E_{\mathcal{H}})$ be a tree rooted at r. Let p be the number of the elements in G, i.e., $p = |V_{\mathcal{H}} \cup E_{\mathcal{H}}|$. For each vertex $v \in V_{\mathcal{H}}$, the *level* of v is defined as $\lambda(v) = d_{\mathcal{H}}(v, r)$, and for each edge $e = (u, v)$ of G, the *level* of e is defined as $\lambda(e) = \frac{1}{2}(\lambda(u) + \lambda(v))$. For any two distinct vertices $u, v \in V_{\mathcal{H}}$, u is called the *parent* of v if $(u, v) \in E_{\mathcal{H}}$ and $\lambda(u) = \lambda(v) - 1$. If u is the parent of v, then v is called a *child* of u. A *level transversal ordering* of $\mathcal{H}$ is an ordering $x_1, x_2, \ldots, x_p$ of the elements of $V_{\mathcal{H}} \cup E_{\mathcal{H}}$ arranged in non-increasing levels of them. Clearly, x_p is the root.

A level transversal ordering of a rooted tree can be computed by breadth first search in $O(n)$ time. Aleš and Bačík [3] showed that a level transversal ordering of a rooted tree $\mathcal{H}$ is a strong elimination ordering of the vertices in $T(\mathcal{H})$.

Algorithm PMD$(\mathcal{H}, \mathcal{P}, I_1, \ell, d))$

for $i = 1$ to p **do**
  if $t(x_i) = F_r$ **then**
    $f(x_i) \leftarrow I_1 + (\ell - 1) \cdot d$;
  else
    $f(x_i) \leftarrow t(x_i)$;
  end if
end for
for $i = 1$ to p **do**
  if $k(x_i) > f(N_G^m[x_i])$ **then**
    stop and return the infeasibility of the problem;
  end if
end for
for $i = 1$ to p **do**
  if $t(x_i) = F_r$ **then**
    $M \leftarrow \min\{f(N_G^m[x_j]) - k(x_j) | x_j \in N_G^m[x_i]\}$;
    $f(x_i) \leftarrow \max\{I_1, I_1 + (\lceil \ell - \frac{M}{d} \rceil - 1) \cdot d\}$;
  end if
end for
return the function f;

We give Algorithm PMD to find a $\mathcal{P}$-mixed dominating function of a rooted tree of minimum weight. The algorithm takes $\mathcal{H}$, $\mathcal{P}$, I_1, ℓ, and d as inputs. Input $\mathcal{H}$ represents a rooted tree $\mathcal{H}$ with a level transversal ordering $x_1, x_2, \ldots, x_p$, where $p = |V(\mathcal{H}) \cup E(\mathcal{H})|$. Inputs ℓ, d, I_1 are fixed integers and $\ell, d > 0$. Let $F_r = I_1 + \ell d$. The weight set Y is assumed to be the set $\{I_1, I_1 + d, I_1 + 2d, \ldots, I_1 + (\ell - 1) \cdot d\}$. Input $\mathcal{P}$ represents a labeling function which assigns to every element $x \in V(\mathcal{H}) \cup E(\mathcal{H})$ a label $\mathcal{P}(x) = (t(x), k(x))$, where $t(x) \in Y \cup \{F_r\}$ and $k(x)$ is a fixed integer.

It can be easily verified that the running time of Algorithm PMD is $O(n^2)$. The correctness of Algorithm PMD can be proved by the arguments similar to those for proving the correctness of the algorithm for the labeled domination on strongly chordal graphs [13].

3 A linear-time algorithm for $\mathcal{P}$-mixed domination in rooted trees

Based on Algorithm PMD, we develop an algorithm, called IPMD, to solve the $\mathcal{P}$-mixed domination problem for rooted trees in $O(n)$ time. Algorithm IPMD contains four procedures: Initialization, Verification, MComputation, and Update. We show how to design these four procedures in this section.

Algorithm IPMD$(\mathcal{H}, \mathcal{P}, I_1, \ell, d))$

Initialization$(\mathcal{H}, \mathcal{P}, I_1, \ell, d))$;
Verification(f);
for $i = 1$ to p **do**
  if $t(x_i) = F_r$ **then**
    MComputation(x_i, f);
    $f(x_i) \leftarrow \max\{I_1, I_1 + (\lceil \ell - \frac{M}{d} \rceil - 1) \cdot d\}$;
    Update(x_i, f);
  end if
end for
return the function f;

Throughout the section, we use $\mathcal{H} = (V_{\mathcal{H}}, E_{\mathcal{H}})$ to denote a rooted tree with a level transversal ordering $x_1, x_2, \ldots, x_p$, where $p = |V_{\mathcal{H}} \cup E_{\mathcal{H}}|$. If an element x_i in $\mathcal{H}$ is an edge, let $x_i = (x_{i_1}, x_{i_2})$ and let x_{i_2} be the parent of x_{i_1}. Clearly, $i_1 < i_2$ and $\lambda(i_1) > \lambda(i_2)$.

Suppose that Y is a subset of real numbers and $f : V_{\mathcal{H}} \cup E_{\mathcal{H}} \to Y$ is a function of $\mathcal{H}$ which assigns a value of Y to an element x in $\mathcal{H}$. For each vertex x_i of $\mathcal{H}$, we let (1) $v_f(x_i) = f(N_{\mathcal{H}}[x_i])$, (2) $e_f(x_i) = f(E_{\mathcal{H}}(x_i))$, (3)$n_f(x_i) = \min\{f(N_{\mathcal{H}}^m[x_j]) - k(x_j) \mid x_j \in N_{\mathcal{H}}[x_i] \text{ and } j \le i\}$, and (4)$m_f(x_i) = \min\{f(N_{\mathcal{H}}^m[x_j]) - k(x_j) \mid x_j \in E_{\mathcal{H}}(x_i) \text{ and } j < i\}$.

Lemma 1. *Suppose that Y is a subset of real numbers. Let $f : V_{\mathcal{H}} \cup E_{\mathcal{H}} \to Y$ be a function of $\mathcal{H}$. The following statements are true.*

(1) If an element x_i is a vertex of $\mathcal{H}$, then $f(N_{\mathcal{H}}^m[x_i]) = v_f(x_i) + e_f(x_i)$.

(2) If an element x_i is an edge of $\mathcal{H}$, then $f(N_{\mathcal{H}}^m[x_i]) = e_f(x_{i_1}) + e_f(x_{i_2}) - f(x_i) + f(x_{i_1}) + f(x_{i_2})$.

Proof. The statements can be easily verified according to the definitions. □

We design a procedure, called Initialization, to initialize a function f of $\mathcal{H}$ and to set the values of $v_f(x_i)$, $e_f(x_i)$, $n_f(x_i)$, and $m_f(x_i)$ for each vertex x_i in $\mathcal{H}$ according to the initialized function f and their definitions. These values will be used for designing the procedures Verification, MComputation, and Update to make them more efficiently.

Procedure INITIALIZATION($\mathcal{H}, \mathcal{P}, I_1, \ell, d$)

1: **for** $i = 1$ to p **do**
2: **if** $t(x_i) = F_r$ **then**
3: $f(x_i) \leftarrow I_1 + (\ell - 1) \cdot d$;
4: **else**
5: $f(x_i) \leftarrow t(x_i)$;
6: **end if**
7: **end for**
8: **for** each vertex $x_i \in V_{\mathcal{H}}$ **do**
9: $v_f(x_i) \leftarrow f(N_{\mathcal{H}}[x_i])$; $e_f(x_i) \leftarrow f(E_{\mathcal{H}}(x_i))$;
10: **end for**
11: **for** each vertex $x_i \in V_{\mathcal{H}}$ **do**
12: $n_f(x_i) \leftarrow \{f(N^m_{\mathcal{H}}[x_j]) - k(x_j) \mid x_j \in N_{\mathcal{H}}[x_i]$ and $j \le i\}$;
13: $m_f(x_i) \leftarrow \{f(N^m_{\mathcal{H}}[x_j]) - k(x_j) \mid x_j \in E_{\mathcal{H}}(x_i)$ and $j < i\}$;
14: **end for**

Theorem 1. *Steps 1–7 of Procedure* INITIALIZATION *are equivalent to Steps 1–7 of Algorithm PMD, and the running time of this procedure is* $O(n)$.

Proof. It can be easily observed that Steps 1–7 of Procedure INITIALIZATION are equivalent to Steps 1–7 of Algorithm PMD.

Since $\mathcal{H}$ is a tree, $p = |V_{\mathcal{H}} \cup E_{\mathcal{H}}| = 2n-1$. Steps 1–7 can be done in $O(n)$ time. For any vertex x_i of $\mathcal{H}$, $|N_{\mathcal{H}}[x_i]| = deg_{\mathcal{H}}(x_i) + 1$ and $|E_{\mathcal{H}}(x_i)| = deg_{\mathcal{H}}(x_i)$. Therefore, $v_f(x_i)$ and $e_f(x_i)$ can be computed in $O(deg_{\mathcal{H}}(x_i) + 1)$ time for each vertex $x_i \in V_{\mathcal{H}}$.

Since the values of $v_f(x_i)$ and $e_f(x_i)$ have been computed in Steps 8–10 for each vertex $x_i \in V_{\mathcal{H}}$, by Lemma 1, we know that $f(N^m_{\mathcal{H}}[x_j])$ can be computed in $O(1)$ time for each element x_j in $\mathcal{H}$. Therefore, $n_f(x_i)$ and $m_f(x_i)$ can be computed in $O(deg_{\mathcal{H}}(x_i)+1)$ time for each vertex $x_i \in V_{\mathcal{H}}$. The total amount of time to run Steps 8–14 is

$$O\left(\sum_{x_i \in V_{\mathcal{H}}} (deg_{\mathcal{H}}(x_i) + 1)\right) = O(n + m) = O(n).$$

Hence, the running time of Procedure INITIALIZATION is $O(n)$. □

Based on Lemma 1, we design Procedure VERIFICATION to check whether the function f, initialized by Procedure INITIALIZATION, is a $\mathcal{P}$-mixed dominating function of $\mathcal{H}$. We have the following theorem.

Theorem 2. *The steps of* Procedure VERIFICATION *are equivalent to Steps 8–12 of Algorithm PMD, and the running time of this procedure is* $O(n)$.

Procedure VERIFICATION(f)

1: **for** $i = 1$ to p **do**
2: **if** $x_i \in V_{\mathcal{H}}$ and $k(x_i) > v_f(x_i) + e_f(x_i)$ or $x_i \in E_{\mathcal{H}}$ and $k(x_i) > e_f(x_{i_1}) + e_f(x_{i_2}) - f(x_i) + f(x_{i_1}) + f(x_{i_2})$ **then**
3: stop and return the infeasibility of the problem;
4: **end if**
5: **end for**

For each iteration of Steps 13–18, Algorithm PMD computes a value, M, in Step 15 if the condition for the if-statement in Step 14 is true. In Algorithm IPMD, we design a procedure called MCOMPUTATION to compute M for each element x_i in $\mathcal{H}$ with $t(x_i) = F_r$. Since x_i is either a vertex or an edge of $\mathcal{H}$, Procedure MCOMPUTATION deals with the case for $x_i \in V_{\mathcal{H}}$ in Steps 1–10 and deals with the case for $x_i \in E_{\mathcal{H}}$ in Steps 11–21. Steps 1–10 of the procedure are based on Lemma 2 and Steps 11–21 are based on Lemma 3.

Procedure MCOMPUTATION(x_i, f)

1: **if** x_i is a vertex of $\mathcal{H}$ **then**
2: **if** x_i is the root of $\mathcal{H}$ **then**
3: $M \leftarrow \min\{n_f(x_i), m_f(x_i)\}$;
4: **else**
5: let $x_{i'}$ be the parent of x_i and let $x_t = (x_i, x_{i'})$;
6: $m_1 \leftarrow v_f(x_{i'}) + e_f(x_{i'}) - k(x_{i'})$;
7: $m_2 \leftarrow e_f(x_i) + e_f(x_{i'}) - f(x_t) + f(x_i) + f(x_{i'}) - k(x_t)$;
8: $M \leftarrow \min\{n_f(x_i), m_f(x_i), m_1, m_2\}$;
9: **end if**
10: **end if**
11: **if** x_i is an edge of $\mathcal{H}$ **then**
12: let $x_i = (x_{i_1}, x_{i_2})$ of $\mathcal{H}$, where x_{i_2} is the parent of x_{i_1};
13: $m_1 \leftarrow \min\{v_f(x_{i_1}) + e_f(x_{i_1}) - k(x_{i_1}), v_f(x_{i_2}) + e_f(x_{i_2}) - k(x_{i_2})\}$;
14: **if** x_{i_2} is the root of $\mathcal{H}$ **then**
15: $M \leftarrow \min\{m_f(x_{i_1}), m_f(x_{i_2}), m_1\}$;
16: **else**
17: let $x_{i'_2}$ be the parent of x_{i_2} and let $x_t = (x_{i_2}, x_{i'_2})$;
18: $m_2 \leftarrow e_f(x_{i_2}) + e_f(x_{i'_2}) - f(x_t) + f(x_{i_2}) + f(x_{i'_2}) - k(x_t)$;
19: $M \leftarrow \min\{m_f(x_{i_1}), m_f(x_{i_2}), m_1, m_2\}$;
20: **end if**
21: **end if**

Lemma 2. *Assume that f is a $\mathcal{P}$-mixed dominating function of $\mathcal{H}$. Let x_i be a vertex of $\mathcal{H}$ and let $M = \min\{f(N^m_{\mathcal{H}}[x_j]) - k(x_j) \mid x_j \in N^m_{\mathcal{H}}[x_i]\}$. The following statements are true.*

(1) Suppose that x_i is the root of $\mathcal{H}$. Then, $M = \min\{n_f(x_i), m_f(x_i)\}$.

(2) Suppose that x_i is not the root of $\mathcal{H}$. Let $x_{i'}$ be the parent of x_i and let $x_t = (x_i, x_{i'})$. Let $m_1 = v_f(x_{i'}) + e_f(x_{i'}) - k(x_{i'})$ and let $m_2 = e_f(x_i) + e_f(x_{i'}) - f(x_t) + f(x_i) + f(x_{i'}) - k(x_t)$. Then, $M = \min\{n_f(x_i), m_f(x_i), m_1, m_2\}$.

Proof. (1) By definition, $N^m_{\mathcal{H}}[x_i] = N_{\mathcal{H}}[x_i] \cup E_{\mathcal{H}}(x_i)$. Since x_i is the root of $\mathcal{H}$, $j \le i$ for any vertex $x_j \in N_{\mathcal{H}}[x_i]$, and $j < i$ for any edge $x_j \in E_{\mathcal{H}}(x_i)$. We have $M = \min\{f(N^m_{\mathcal{H}}[x_j]) - k(x_j) \mid x_j \in N^m_{\mathcal{H}}[x_i]\} = \min\{f(N^m_{\mathcal{H}}[x_j]) - k(x_j) \mid x_j \in N_{\mathcal{H}}[x_i] \cup E_{\mathcal{H}}(x_i)\} = \min\{n_f(x_i), m_f(x_i)\}$.

(2) Note that $N^m_{\mathcal{H}}[x_i] = N_{\mathcal{H}}[x_i] \cup E_{\mathcal{H}}(x_i)$. For each element $x_j \in N^m_{\mathcal{H}}[x_i] \setminus \{x_t, x_{i'}\}$, the index j is smaller than or equal to i. Therefore, $\min\{f(N^m_{\mathcal{H}}[x_j]) - k(x_j) \mid x_j \in N^m_{\mathcal{H}}[x_i] \setminus \{x_t, x_{i'}\}\} = \min\{n_f(x_i), m_f(x_i)\}$. We have $M = \min\{f(N^m_{\mathcal{H}}[x_j]) - k(x_j) \mid x_j \in N^m_{\mathcal{H}}[x_i]\} = \min\{n_f(x_i), m_f(x_i), f(N^m_{\mathcal{H}}[x_{i'}]) - k(x_{i'}), f(N^m_{\mathcal{H}}[x_t]) - k(x_t)\}$.

By Lemma 1, $f(N^m_{\mathcal{H}}[x_{i'}]) = v_f(x_{i'}) + e_f(x_{i'})$ and $f(N^m_{\mathcal{H}}[x_t]) = e_f(x_i) + e_f(x_{i'}) - f(x_t) + f(x_i) + f(x_{i'})$. Therefore, $M = \min\{n_f(x_i), m_f(x_i), m_1, m_2\}$. □

Lemma 3. *Assume that f is a $\mathcal{P}$-mixed dominating function of $\mathcal{H}$. Let x_i be an edge (x_{i_1}, x_{i_2}) of $\mathcal{H}$, where x_{i_2} is the parent of x_{i_1}. Let $M = \min\{f(N^m_{\mathcal{H}}[x_j]) - k(x_j) \mid x_j \in N^m_{\mathcal{H}}[x_i]\}$ and let $m_1 = \min\{v_f(x_{i_1}) + e_f(x_{i_1}) - k(x_{i_1}), v_f(x_{i_2}) + e_f(x_{i_2}) - k(x_{i_2})\}$. The following statements are true.*

(1) Suppose that x_{i_2} is the root of $\mathcal{H}$. Then, $M = \min\{m_f(x_{i_1}), m_f(x_{i_2}), m_1\}$.

(2) Suppose that x_{i_2} is not the root of $\mathcal{H}$. Let $x_{i'_2}$ be the parent of x_{i_2} and $x_t = (x_{i_2}, x_{i'_2})$. Let $m_2 = e_f(x_{i_2}) + e_f(x_{i'_2}) - f(x_t) + f(x_{i_2}) + f(x_{i'_2}) - k(x_t)$. Then, $M = \min\{m_f(x_{i_1}), m_f(x_{i_2}), m_1, m_2\}$.

Proof. (1) By definition, $N^m_{\mathcal{H}}[x_i] = E_{\mathcal{H}}(x_{i_1}) \cup E_{\mathcal{H}}(x_{i_2}) \cup \{x_{i_1}, x_{i_2}\}$. Note that $E_{\mathcal{H}}(x_{i_1}) \cap E_{\mathcal{H}}(x_{i_2}) = \{x_i\}$. Since x_{i_2} is the root of $\mathcal{H}$, $j < i_1$ for any element $x_j \in E_{\mathcal{H}}(x_{i_1}) \setminus \{x_i\}$, and $j < i_2$ for every element $x_j \in E_{\mathcal{H}}(x_{i_2})$. We have $M = \min\{f(N^m_{\mathcal{H}}[x_j]) - k(x_j) \mid x_j \in N^m_{\mathcal{H}}[x_i]\} = \min\{f(N^m_{\mathcal{H}}[x_j]) - k(x_j) \mid x_j \in E_{\mathcal{H}}(x_{i_1}) \cup E_{\mathcal{H}}(x_{i_2}) \cup \{x_{i_1}, x_{i_2}\}\} = \min\{m_f(x_{i_1}), m_f(x_{i_2}), f(N^m_{\mathcal{H}}[x_{i_1}]) - k(x_{i_1}), f(N^m_{\mathcal{H}}[x_{i_2}]) - k(x_{i_2})\}$. By Lemma 1, we know that $f(N^m_{\mathcal{H}}[x_{i_1}]) = v_f(x_{i_1}) + e_f(x_{i_1})$ and $f(N^m_{\mathcal{H}}[x_{i_2}]) = v_f(x_{i_2}) + e_f(x_{i_2})$. Hence, $M = \min\{m_f(x_{i_1}), m_f(x_{i_2}), m_1\}$.

(2) By definition, $N^m_{\mathcal{H}}[x_i] = E_{\mathcal{H}}(x_{i_1}) \cup E_{\mathcal{H}}(x_{i_2}) \cup \{x_{i_1}, x_{i_2}\}$. Note that $E_{\mathcal{H}}(x_{i_1}) \cap E_{\mathcal{H}}(x_{i_2}) = \{x_i\}$. For each element $x_j \in N^m_{\mathcal{H}}[x_i] \setminus \{x_{i_1}, x_{i_2}, x_t\}$, $j < i_1$ or $j < i_2$. Therefore, $\min\{f(N^m_{\mathcal{H}}[x_j]) - k(x_j) \mid x_j \in N^m_{\mathcal{H}}[x_i] \setminus \{x_{i_1}, x_{i_2}, x_t\}\} = \min\{m_f(x_{i_1}), m_f(x_{i_2})\}$. We have $M = \min\{f(N^m_{\mathcal{H}}[x_j]) - k(x_j) \mid x_j \in N^m_{\mathcal{H}}[x_i]\} = \min\{m_f(x_{i_1}), m_f(x_{i_2}), f(N^m_{\mathcal{H}}[x_{i_1}]) - k(x_{i_1}), f(N^m_{\mathcal{H}}[x_{i_2}]) - k(x_{i_2}), f(N^m_{\mathcal{H}}[x_t]) - k(x_t)\}$. By Lemma 1, we know that $f(N^m_{\mathcal{H}}[x_{i_1}]) = v_f(x_{i_1}) + e_f(x_{i_1})$, $f(N^m_{\mathcal{H}}[x_{i_2}]) = v_f(x_{i_2}) + e_f(x_{i_2})$, and $f(N^m_{\mathcal{H}}[x_t]) = e_f(x_{i_2}) + e_f(x_{i'_2}) - f(x_t) + f(x_{i_2}) + f(x_{i'_2})$. Hence, $M = \min\{m_f(x_{i_1}), m_f(x_{i_2}), m_1, m_2\}$. □

Theorem 3. *Procedure* MCOMPUTATION *is equivalent to Step 15 of Algorithm PMD, and the running time of this procedure is $O(1)$.*

Proof. It can be easily verified that the running time of Procedure MCOMPUTATION is $O(1)$.

For any element x_i in $\mathcal{H}$, it is either a vertex or an edge. By Lemmas 2 and 3, Procedure MCOMPUTATION is equivalent to Step 15 of Algorithm PMD. □

By Lemmas 2–3, we design Procedure MCOMPUTATION to compute M. However, for each i-th iteration of Steps 3–9, Algorithm IPMD replaces the value of $f(x_i)$ with $\max\{I_1, I_1 + (\lceil \ell - \frac{M}{d} \rceil - 1) \cdot d\}$ if $t(x_i) = F_r$. We therefore design a procedure called UPDATE to make sure that both at the beginning and at the end of each iteration of Steps 3–9, the function f satisfies the following conditions for each vertex x_j in $\mathcal{H}$: (1) $v_f(x_j) = f(N_{\mathcal{H}}[x_j])$, (2) $e_f(x_j) = f(E_{\mathcal{H}}(x_j))$, (3)$n_f(x_j) = \min\{f(N^m_{\mathcal{H}}[x_k]) - k(x_k) \mid x_k \in N_{\mathcal{H}}[x_j] \text{ and } k \le j\}$, and (4)$m_f(x_j) = \min\{f(N^m_{\mathcal{H}}[x_k]) - k(x_k) \mid x_k \in E_{\mathcal{H}}(x_j) \text{ and } k < j\}$.

Note that Algorithm IPMD changes only the value of $f(x_i)$ at each i-th iteration of Steps 3–9. Since x_i is either a vertex or an edge of $\mathcal{H}$, Procedure UPDATE deals with the case for $x_i \in V_{\mathcal{H}}$ in Steps 2–29 and deals with the case for $x_i \in E_{\mathcal{H}}$ in Steps 30–55. Steps 2–29 of the procedure are based on Lemma 4 and Steps 30–55 are based on Lemma 5.

Procedure UPDATE(x_i, f)

$\mathcal{B} \leftarrow \max\{I_1, I_1 + (\lceil \ell - \frac{M}{d} \rceil - 1) \cdot d\}$;
if x_i is a vertex of $\mathcal{H}$ **then**
  for each vertex $x_j \in N_{\mathcal{H}}[x_i]$ **do**
    $v_f(x_j) \leftarrow v_f(x_j) - (I_1 + (\ell - 1) \cdot d) + \mathcal{B}$;
  end for
  if there exists a vertex $x_{i''}$ such that $x_{i''}$ is the parent of the parent of x_i **then**
    $W \leftarrow N_{\mathcal{H}}[x_i] \cup \{x_{i''}\}$;
  else
    $W \leftarrow N_{\mathcal{H}}[x_i]$;
  end if
  for each vertex $x_j \in W$ **do**
    if x_j is a child of x_i **then**
      $n_f(x_j) \leftarrow \min\{n_f(x_j), v_f(x_j) + e_f(x_j) - k(x_j)\}$;
    end if
    if x_j is the vertex x_i **then**
      $n_f(x_j) \leftarrow n_f(x_j) - (I_1 + (\ell - 1) \cdot d) + \mathcal{B}$;
      $m_f(x_j) \leftarrow m_f(x_j) - (I_1 + (\ell - 1) \cdot d) + \mathcal{B}$;
    end if
    if x_j is the parent of x_i **then**
      let $x_t = (x_i, x_j)$;
      $n_f(x_j) \leftarrow \min\{n_f(x_j), v_f(x_j) + e_f(x_j) - k(x_j), v_f(x_i) + e_f(x_i) - k(x_i)\}$;
      $m_f(x_j) \leftarrow \min\{m_f(x_j), e_f(x_i) + e_f(x_j) - f(x_t) + \mathcal{B} + f(x_j) - k(x_t)\}$;
    end if
    if x_j is the parent of the parent of x_i **then**
      let $x_{i'}$ be the parent of x_i;
      $n_f(x_j) \leftarrow \min\{n_f(x_j), v_f(x_{i'}) + e_f(x_{i'}) - k(x_{i'})\}$;
    end if
  end for
end if
if x_i is an edge of $\mathcal{H}$ **then**
  let $x_i = (x_{i_1}, x_{i_2})$ be an edge of $\mathcal{H}$, where x_{i_2} is the parent of x_{i_1};
  for each vertex $x_j \in \{x_{i_1}, x_{i_2}\}$ **do**
    $e_f(x_j) \leftarrow e_f(x_j) - (I_1 + (\ell - 1) \cdot d) + \mathcal{B}$;
  end for
  if there exists a vertex $x_{i'_2}$ such that $x_{i'_2}$ is the parent of x_{i_2} **then**
    $W \leftarrow \{x_{i_1}, x_{i_2}, x_{i'_2}\}$;
  else
    $W \leftarrow \{x_{i_1}, x_{i_2}\}$;
  end if
  for each vertex $x_j \in W$ **do**
    if $x_j = x_{i_1}$ **then**
      $n_f(x_j) \leftarrow \min\{n_f(x_j), v_f(x_j) + e_f(x_j) - k(x_j)\}$;
      $m_f(x_j) \leftarrow m_f(x_j) - (I_1 + (\ell - 1) \cdot d) + \mathcal{B}$;
    end if
    if $x_j = x_{i_2}$ **then**
      $n_f(x_j) \leftarrow \min\{n_f(x_j), v_f(x_{i_1}) + e_f(x_{i_1}) - k(x_{i_1}), v_f(x_j) + e_f(x_j) - k(x_j)\}$;
      $m_f(x_j) \leftarrow m_f(x_j) - (I_1 + (\ell - 1) \cdot d) + \mathcal{B}$;
    end if
    if x_j is the parent of x_{i_2} **then**
      let $x_t = (x_{i_2}, x_j)$;
      $n_f(x_j) \leftarrow \min\{n_f(x_j), v_f(x_{i_2}) + e_f(x_{i_2}) - k(x_{i_2})\}$;
      $m_f(x_j) \leftarrow \min\{m_f(x_j), e_f(x_{i_2}) + e_f(x_j) - f(x_t) + f(x_{i_2}) + f(x_j)\}$;
    end if
  end for
end if

Lemma 4. *Assume that Algorithm IPMD runs and does not stop in Step 2. Let* $Y = \{I_1, I_1 + d, \ldots, I_1 + (\ell - 1) \cdot d\}$ *and let* $f : V_{\mathcal{H}} \cup E_{\mathcal{H}} \to Y$ *be the function of* $\mathcal{H}$ *at the beginning of the i-th iteration of Steps 3–9. Let* x_i *be a vertex of* $\mathcal{H}$ *with* $t(x_i) = F_r$ *and let* $f' : V_{\mathcal{H}} \cup E_{\mathcal{H}} \to Y$ *be a function of* $\mathcal{H}$ *such that* $f'(x_i) = \mathcal{B} = \max\{I_1, I_1 + (\lceil \ell - \frac{M}{d} \rceil - 1) \cdot d\}$ *and* $f'(x_j) = f(x_j)$ *for every element* $x_j \in V_{\mathcal{H}} \cup E_{\mathcal{H}} \setminus \{x_i\}$. *The following statements are true.*

(1) For each vertex $x_j \in V_{\mathcal{H}}$, $e_{f'}(x_j) = e_f(x_j)$.

(2) For each vertex $x_j \in V_{\mathcal{H}}$, $v_{f'}(x_j) = v_f(x_j) - (I_1 + (\ell - 1) \cdot d) + \mathcal{B}$ *if* $x_j \in N_{\mathcal{H}}[x_i]$. *Otherwise,* $v_{f'}(x_j) = v_f(x_j)$.

(3) If there exists a vertex $x_{i''}$ *such that* $x_{i''}$ *is the parent of the parent of* x_i, *then let* $W = N_{\mathcal{H}}[x_i] \cup \{x_{i''}\}$. *Otherwise, let* $W = N_{\mathcal{H}}[x_i]$. *For each vertex* $x_j \in V_{\mathcal{H}} \setminus W$, $n_{f'}(x_j) = n_f(x_j)$ *and* $m_{f'}(x_j) = m_f(x_j)$.

(4) Let x_j *be a vertex in* W. *The following statements are true.*

  (4.1) Suppose that x_j *is a child of* x_i. *Then,* $m_{f'}(x_j) = m_f(x_j)$ *and* $n_{f'}(x_j) = \min\{n_f(x_j), v_{f'}(x_j) + e_f(x_j) - k(x_j)\}$.

  (4.2) Suppose that x_j *is the vertex* x_i. *Then,* $m_{f'}(x_j) = m_f(x_j) - (I_1 + (\ell - 1) \cdot d) + \mathcal{B}$ *and* $n_{f'}(x_j) = n_f(x_j) - (I_1 + (\ell - 1) \cdot d) + \mathcal{B}$.

  (4.3) Suppose that x_j *is the parent of* x_i. *Let* $x_t = (x_i, x_j)$. *Then,* $n_{f'}(x_j) = \min\{n_f(x_j), v_{f'}(x_j) +$

$e_f(x_j) - k(x_j), v_{f'}(x_i) + e_f(x_i) - k(x_i)\}$, *and* $m_{f'}(x_j) = \min\{m_f(x_j), e_f(x_i) + e_f(x_j) - f(x_t) + \mathcal{B} + f(x_j) - k(x_t)\}$.

(4.4) Suppose that x_j is the parent of the parent of x_i. Let $x_{i'}$ be the parent of x_i. Then, $m_{f'}(x_j) = m_f(x_j)$ and $n_{f'}(x_j) = \min\{n_f(x_j), v_{f'}(x_{i'}) + e_f(x_{i'}) - k(x_{i'})\}$.

Proof. (1) Note that $f'(x_k) = f(x_k)$ for every element $x_k \in V_{\mathcal{H}} \cup E_{\mathcal{H}} \setminus \{x_i\}$. Let x_j be a vertex of $\mathcal{H}$. By definition, $e_{f'}(x_j) = f'(E_{\mathcal{H}}(x_j))$. Since x_i is a verex, $x_i \notin E_{\mathcal{H}}(x_j)$ and thus $e_{f'}(x_j) = e_f(x_j)$.

(2) Let x_j be a vertex of $\mathcal{H}$. By definition, $v_{f'}(x_j) = f'(N_{\mathcal{H}}[x_j])$. If $x_i \notin N_{\mathcal{H}}[x_j]$, then $v_{f'}(x_j) = v_f(x_j)$. We now suppose that $x_i \in N_{\mathcal{H}}[x_j]$. Note that $f(x_i) = I_1 + (\ell - 1) \cdot d$ at the beginning of the i-th iteration of Steps 3–9 in Algorithm IPMD. Then, $f'(x_i) = f(x_i) - (I_1 + (\ell - 1) \cdot d) + \mathcal{B}$. Since $f'(x_k) = f(x_k)$ for every element $x_k \in V_{\mathcal{H}} \cup E_{\mathcal{H}} \setminus \{x_i\}$, we have $v_{f'}(x_j) - f'(x_i) = v_f(x_j) - f(x_i)$. Hence, $v_{f'}(x_j) = v_f(x_j) - (I_1 + (\ell - 1) \cdot d) + \mathcal{B}$.

(3) Let x_j be a vertex in $V_{\mathcal{H}} \setminus W$. Note that $n_{f'}(x_j) = \min\{f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) \mid x_k \in N_{\mathcal{H}}[x_j] \text{ and } k \le j\}$ and $m_{f'}(x_j) = \min\{f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) \mid x_k \in E_{\mathcal{H}}(x_j) \text{ and } k < j\}$. By the definition of W, it can be easily verified that for any element $x_k \in N_{\mathcal{H}}[x_j] \cup E_{\mathcal{H}}(x_j)$ with $k \le j$, $N^m_{\mathcal{H}}[x_k]$ does not contain the vertex x_i. Hence, $n_{f'}(x_j) = n_f(x_j)$ and $m_{f'}(x_j) = m_f(x_j)$.

(4) For each vertex $x_j \in W$, by definition, $n_{f'}(x_j) = \min\{f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) \mid x_k \in N_{\mathcal{H}}[x_j] \text{ and } k \le j\}$ and $m_{f'}(x_j) = \min\{f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) \mid x_k \in E_{\mathcal{H}}(x_j) \text{ and } k < j\}$. We consider the following four cases.

Case 1: x_j is a child of x_i. It can be easily verified that for any edge $x_k \in E_{\mathcal{H}}(x_j)$ with $k < j$, $N^m_{\mathcal{H}}[x_k]$ does not contain the vertex x_i. Therefore, $f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) = f(N^m_{\mathcal{H}}[x_k]) - k(x_k)$. We have $m_{f'}(x_j) = m_f(x_j)$.

We now consider a vertex $x_k \in N_{\mathcal{H}}[x_j]$ with $k \le j$. If $k < j$, then $N^m_{\mathcal{H}}[x_k]$ does not contain the vertex x_i. Therefore, $f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) = f(N^m_{\mathcal{H}}[x_k]) - k(x_k)$. If $k = j$, then $f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) = f'(N^m_{\mathcal{H}}[x_j]) - k(x_j)$ and $N^m_{\mathcal{H}}[x_k]$ contains the vertex x_i. By Lemma 1, and Statements (1) and (2) of this lemma, $f'(N^m_{\mathcal{H}}[x_j]) - k(x_j) = v_{f'}(x_j) + e_{f'}(x_j) - k(x_j) = v_{f'}(x_j) + e_f(x_j) - k(x_j) \le v_f(x_j) + e_f(x_j) - k(x_j) = f(N^m_{\mathcal{H}}[x_j]) - k(x_j)$. Therefore, $n_{f'}(x_j) = \min\{f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) \mid x_k \in N_{\mathcal{H}}[x_j] \text{ and } k \le j\} = \min\{n_f(x_j), f'(N^m_{\mathcal{H}}[x_j]) - k(x_j)\} = \min\{n_f(x_j), v_{f'}(x_j) + e_f(x_j) - k(x_j)\}$.

Case 2: x_j is the vertex x_i. For any element $x_k \in N_{\mathcal{H}}[x_j] \cup E_{\mathcal{H}}(x_j)$ with $k \le j$, $N^m_{\mathcal{H}}[x_k]$ contains the vertex x_i. Note that $f(x_i) = I_1 + (\ell - 1) \cdot d$ at the beginning of the i-th iteration of Steps 3–9 in Algorithm IPMD. We have $f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) = f(N^m_{\mathcal{H}}[x_k]) - (I_1 + (\ell - 1) \cdot d) + \mathcal{B} - k(x_k)$ for any element $x_k \in N_{\mathcal{H}}[x_j] \cup E_{\mathcal{H}}(x_j)$ with $k \le j$. Hence, $m_{f'}(x_j) = m_f(x_j) - (I_1 + (\ell - 1) \cdot d) + \mathcal{B}$ and $n_{f'}(x_j) = n_f(x_j) - (I_1 + (\ell - 1) \cdot d) + \mathcal{B}$.

Case 3: x_j is the parent of x_i. Clearly, $i < j$. Let x_k be a vertex in $N_{\mathcal{H}}[x_j]$ with $k \le j$. If $k \ne i, j$, then $N^m_{\mathcal{H}}[x_k]$ does not contain the vertex x_i. Therefore, $f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) = f(N^m_{\mathcal{H}}[x_k]) - k(x_k)$. If $k = i$ or $k = j$, then $N^m_{\mathcal{H}}[x_k]$ contains the vertex x_i. By Lemma 1, we know that $f'(N^m_{\mathcal{H}}[x_j]) - k(x_j) = v_{f'}(x_j) + e_{f'}(x_j) - k(x_j)$, and $f'(N^m_{\mathcal{H}}[x_i]) - k(x_i) = v_{f'}(x_i) + e_{f'}(x_i) - k(x_i)$. By Statements (1) and (2) of this lemma, $v_{f'}(x_i) + e_{f'}(x_i) - k(x_i) = v_{f'}(x_i) + e_f(x_i) - k(x_i) \le v_f(x_i) + e_f(x_i) - k(x_i)$, and $v_{f'}(x_j) + e_{f'}(x_j) - k(x_j) = v_{f'}(x_j) + e_f(x_j) - k(x_j) \le v_f(x_j) + e_f(x_j) - k(x_j)$. Hence, $n_{f'}(x_j) = \min\{n_f(x_j), v_{f'}(x_j) + e_f(x_j) - k(x_j), v_{f'}(x_i) + e_f(x_i) - k(x_i)\}$.

We now consider an edge x_k in $E_{\mathcal{H}}(x_j)$ with $k < j$. Recall that $x_t = (x_i, x_j)$. Clearly, $t < j$. If $k \ne t$, then $N^m_{\mathcal{H}}[x_k]$ does not contain the vertex x_i. Therefore, $f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) = f(N^m_{\mathcal{H}}[x_k]) - k(x_k)$. If $k = t$, then $f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) = f'(N^m_{\mathcal{H}}[x_t]) - k(x_t)$ and $N^m_{\mathcal{H}}[x_k]$ contains the vertex x_i. Note that $f'(x_i) = \mathcal{B}$, and $f'(x) = f(x)$ for every element $x \in V_{\mathcal{H}} \cup E_{\mathcal{H}} \setminus \{x_i\}$. By Lemma 1 and Statement (1) of this lemma, $f'(N^m_{\mathcal{H}}[x_t]) - k(x_t) = e_{f'}(x_i) + e_{f'}(x_j) - f'(x_t) + f'(x_i) + f'(x_j) - k(x_t) = e_f(x_i) + e_f(x_j) - f(x_t) + \mathcal{B} + f(x_j) - k(x_t) \le f(N^m_{\mathcal{H}}[x_t]) - k(x_t)$. Hence, $m_{f'}(x_j) = \min\{f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) \mid x_k \in E_{\mathcal{H}}(x_j) \text{ and } k < j\} = \min\{m_f(x_j), f'(N^m_{\mathcal{H}}[x_t]) - k(x_t)\} = \min\{m_f(x_j), e_f(x_i) + e_f(x_j) - f(x_t) + \mathcal{B} + f(x_j) - k(x_t)\}$.

Case 4: x_j is the parent of the parent $x_{i'}$ of x_i. For any edge $x_k \in E_{\mathcal{H}}(x_j)$ with $k < j$, it can be easily verified that $N^m_{\mathcal{H}}[x_k]$ does not contain the vertex x_i. Therefore, $m_{f'}(x_j) = m_f(x_j)$.

We now consider a vertex $x_k \in N_{\mathcal{H}}[x_j]$ with $k \le j$. Clearly, $x_{i'} \in N_{\mathcal{H}}[x_j]$ and $i' < j$. If $x_k \ne x_{i'}$, then $N^m_{\mathcal{H}}[x_k]$ does not contain the vertex x_i. Therefore, $f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) = f(N^m_{\mathcal{H}}[x_k]) - k(x_k)$. If $x_k = x_{i'}$, then $f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) = f'(N^m_{\mathcal{H}}[x_{i'}]) - k(x_{i'})$ and $N^m_{\mathcal{H}}[x_{i'}]$ contains the vertex x_i. By Lemma 1, and Statements (1) and (2) of this lemma, $f'(N^m_{\mathcal{H}}[x_{i'}]) - k(x_{i'}) = v_{f'}(x_{i'}) + e_{f'}(x_{i'}) - k(x_{i'}) = v_{f'}(x_{i'}) + e_f(x_{i'}) - k(x_{i'}) \le f(N^m_{\mathcal{H}}[x_{i'}]) - k(x_{i'})$. Hence, $n_{f'}(x_j) =$

$\min\{n_f(x_j), v_{f'}(x_{i'}) + e_f(x_{i'}) - k(x_{i'})\}$. □

Lemma 5. *Assume that Algorithm IPMD runs and does not stop in Step 2. Let* $Y = \{I_1, I_1 + d, \ldots, I_1 + (\ell-1)\cdot d\}$ *and let* $f : V_{\mathcal{H}} \cup E_{\mathcal{H}} \to Y$ *be the function of* $\mathcal{H}$ *at the beginning of the* i*-th iteration of Steps 3–9. Let* $x_i = (x_{i_1}, x_{i_2})$ *be an edge of* $\mathcal{H}$ *such that* $t(x_i) = F_r$ *and* x_{i_2} *is the parent of* x_{i_1}*. Let* $f' : V_{\mathcal{H}} \cup E_{\mathcal{H}} \to Y$ *be a function of* $\mathcal{H}$ *such that* $f'(x_i) = \mathcal{B} = \max\{I_1, I_1 + (\lceil \ell - \frac{M}{d} \rceil - 1) \cdot d\}$ *and* $f'(x_j) = f(x_j)$ *for every element* $x_j \in V_{\mathcal{H}} \cup E_{\mathcal{H}} \setminus \{x_i\}$*. The following statements are true.*

(1) For each vertex $x_j \in V_{\mathcal{H}}$*,* $v_{f'}(x_j) = v_f(x_j)$*.*

(2) For each vertex $x_j \in V_{\mathcal{H}}$*,* $e_{f'}(x_j) = e_f(x_j) - (I_1 + (\ell-1)\cdot d) + \mathcal{B}$ *if* $x_j \in \{x_{i_1}, x_{i_2}\}$*. Otherwise,* $e_{f'}(x_j) = e_f(x_j)$*.*

(3) If there exists a vertex $x_{i'_2}$ *such that* $x_{i'_2}$ *is the parent of* x_{i_2}*, then let* $W = \{x_{i_1}, x_{i_2}, x_{i'_2}\}$*. Otherwise, let* $W = \{x_{i_1}, x_{i_2}\}$*. For each vertex* $x_j \in V_{\mathcal{H}} \setminus W$*,* $n_{f'}(x_j) = n_f(x_j)$ *and* $m_{f'}(x_j) = m_f(x_j)$*.*

(4) Let x_j *be a vertex in* W*. The following statements are true.*

(4.1) Suppose that $x_j = x_{i_1}$*. Then,* $n_{f'}(x_j) = \min\{n_f(x_j), v_f(x_j) + e_{f'}(x_j) - k(x_j)\}$*, and* $m_{f'}(x_j) = m_f(x_j) - (I_1 + (\ell-1)\cdot d) + \mathcal{B}$*.*

(4.2) Suppose that $x_j = x_{i_2}$*. Then,* $n_{f'}(x_j) = \min\{n_f(x_j), v_f(x_{i_1}) + e_{f'}(x_{i_1}) - k(x_{i_1}), v_f(x_j) + e_{f'}(x_j) - k(x_j)\}$ *and* $m_{f'}(x_j) = m_f(x_j) - (I_1 + (\ell-1)\cdot d) + \mathcal{B}$*.*

(4.3) Suppose that x_j *is the parent of* x_{i_2}*. Let* $x_t = (x_{i_2}, x_j)$*. Then,* $n_{f'}(x_j) = \min\{n_f(x_j), v_f(x_{i_2}) + e_{f'}(x_{i_2}) - k(x_{i_2})\}$ *and* $m_{f'}(x_j) = \min\{m_f(x_j),\ e_{f'}(x_{i_2}) + e_f(x_j) - f(x_t) + f(x_{i_2}) + f(x_j)\}$*.*

Proof. (1) Note that $f'(x_k) = f(x_k)$ for every element $x_k \in V_{\mathcal{H}} \cup E_{\mathcal{H}} \setminus \{x_i\}$. Let x_j be a vertex of $\mathcal{H}$. By definition, $v_{f'}(x_j) = f'(N_{\mathcal{H}}[x_j])$. Since the edge x_i is not in $N_{\mathcal{H}}[x_j]$, $v_{f'}(x_j) = v_f(x_j)$.

(2) Let x_j be a vertex of $\mathcal{H}$. By definition, $e_{f'}(x_j) = f'(E_{\mathcal{H}}(x_j))$. If $x_i \notin E_{\mathcal{H}}(x_j)$, then $e_{f'}(x_j) = e_f(x_j)$. We now suppose that $x_i \in E_{\mathcal{H}}(x_j)$. Then either $x_j = x_{i_1}$ or $x_j = x_{i_2}$. Note that $f(x_i) = I_1 + (\ell-1)\cdot d$ at the beginning of the i-th iteration of Steps 3–9 in Algorithm IPMD. Then, $f'(x_i) = f(x_i) - (I_1 + (\ell-1)\cdot d) + \mathcal{B}$. Since $f'(x_k) = f(x_k)$ for every element $x_k \in V_{\mathcal{H}} \cup E_{\mathcal{H}} \setminus \{x_i\}$, we have $e_{f'}(x_j) - f'(x_i) = e_f(x_j) - f(x_i)$. Hence, $e_{f'}(x_j) = e_f(x_j) - (I_1 + (\ell-1)\cdot d) + \mathcal{B}$.

(3) Let x_j be a vertex in $V_{\mathcal{H}} \setminus W$. Note that $n_{f'}(x_j) = \min\{f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) \mid x_k \in N_{\mathcal{H}}[x_j] \text{ and } k \le j\}$ and $m_{f'}(x_j) = \min\{f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) \mid x_k \in E_{\mathcal{H}}(x_j) \text{ and } k < j\}$. By the definition of W, it can be easily verified that for any element $x_k \in N_{\mathcal{H}}[x_j] \cup E_{\mathcal{H}}(x_j)$ with $k \le j$, $N^m_{\mathcal{H}}[x_k]$ does not contain the edge x_i. Hence, $n_{f'}(x_j) = n_f(x_j)$ and $m_{f'}(x_j) = m_f(x_j)$.

(4) For each vertex $x_j \in W$, by definition, $n_{f'}(x_j) = \min\{f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) \mid x_k \in N_{\mathcal{H}}[x_j] \text{ and } k \le j\}$ and $m_{f'}(x_j) = \min\{f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) \mid x_k \in E_{\mathcal{H}}(x_j) \text{ and } k < j\}$. We consider the following three cases.

Case 1: $x_j = x_{i_1}$. For any edge $x_k \in E_{\mathcal{H}}(x_j)$ with $k < j$, $N^m_{\mathcal{H}}[x_k]$ contain the edge x_i. Note that $f(x_i) = I_1 + (\ell-1)\cdot d$ at the beginning of the i-th iteration of Steps 3–9 in Algorithm IPMD. Then, $f'(x_i) = f(x_i) - (I_1 + (\ell-1)\cdot d) + \mathcal{B}$. Since $f'(x) = f(x)$ for every element $x \in V_{\mathcal{H}} \cup E_{\mathcal{H}} \setminus \{x_i\}$, $f'(N^m_{\mathcal{H}}[x_k]) - f'(x_i) = f(N^m_{\mathcal{H}}[x_k]) - f(x_i)$. We have $f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) = f(N^m_{\mathcal{H}}[x_k]) - (I_1 + (\ell-1)\cdot d) + \mathcal{B} - k(x_k)$. Therefore, $m_{f'}(x_j) = m_f(x_j) - (I_1 + (\ell-1)\cdot d) + \mathcal{B}$.

We now consider a vertex $x_k \in N_{\mathcal{H}}[x_j]$ with $k \le j$. If $k < j$, then $N^m_{\mathcal{H}}[x_k]$ does not contain the edge x_i. Therefore, $f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) = f(N^m_{\mathcal{H}}[x_k]) - k(x_k)$. If $k = j$, then $f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) = f'(N^m_{\mathcal{H}}[x_j]) - k(x_j)$ and $N^m_{\mathcal{H}}[x_k]$ contains the edge x_i. By Lemma 1, $f'(N^m_{\mathcal{H}}[x_j]) - k(x_j) = v_{f'}(x_j) + e_{f'}(x_j) - k(x_j)$. By Statements (1) and (2) of this lemma, we know that $v_{f'}(x_j) + e_{f'}(x_j) - k(x_j) = v_f(x_j) + e_{f'}(x_j) - k(x_j) \le v_f(x_j) + e_f(x_j) - k(x_j)$. Therefore, $n_{f'}(x_j) = \min\{n_f(x_j), v_f(x_j) + e_{f'}(x_j) - k(x_j)\}$.

Case 2: $x_j = x_{i_2}$. For any vertex $x_k \in N_{\mathcal{H}}[x_j] \setminus \{x_{i_1}, x_j\}$ with $k < j$, $N^m_{\mathcal{H}}[x_k]$ does not contain the edge x_i. Therefore, $f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) = f(N^m_{\mathcal{H}}[x_k]) - k(x_k)$. If $x_k = x_{i_1}$ or $x_k = x_j$, then $N^m_{\mathcal{H}}[x_k]$ contains the edge x_i. By Lemma 1, we know that $f'(N^m_{\mathcal{H}}[x_{i_1}]) - k(x_{i_1}) = v_{f'}(x_{i_1}) + e_{f'}(x_{i_1}) - k(x_{i_1})$, and $f'(N^m_{\mathcal{H}}[x_j]) - k(x_j) = v_{f'}(x_j) + e_{f'}(x_j) - k(x_j)$. By Statements (1) and (2) of this lemma, $v_{f'}(x_{i_1}) + e_{f'}(x_{i_1}) - k(x_{i_1}) = v_f(x_{i_1}) + e_{f'}(x_{i_1}) - k(x_{i_1}) \le v_f(x_{i_1}) + e_f(x_{i_1}) - k(x_{i_1})$, and $v_{f'}(x_j) + e_{f'}(x_j) - k(x_j) = v_f(x_j) + e_{f'}(x_j) - k(x_j) \le v_f(x_j) + e_f(x_j) - k(x_j)$. Hence, $n_{f'}(x_j) = \min\{n_f(x_j),\ v_f(x_j) + e_{f'}(x_j) - k(x_j), v_f(x_{i_1}) + e_{f'}(x_{i_1}) - k(x_{i_1})\}$.

We now consider an edge x_k in $E_{\mathcal{H}}(x_j)$ with $k < j$. Clearly, $x_i \in N^m_{\mathcal{H}}[x_k]$. Note that $f(x_i) = I_1 + (\ell-1)\cdot d$ at the beginning of the i-th iteration of Steps 3–9 in Algorithm IPMD. We have

$f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) = f(N^m_{\mathcal{H}}[x_k]) - (I_1 + (\ell-1)\cdot d) + \mathcal{B} - k(x_k)$. Hence, $m_{f'}(x_j) = m_f(x_j) - (I_1 + (\ell-1)\cdot d) + \mathcal{B}$.

Case 3: x_j is the parent of x_{i_2}. Let x_k be a vertex in $N_{\mathcal{H}}[x_j]$ with $k \le j$. If $k \neq i_2$, then $N^m_{\mathcal{H}}[x_k]$ does not contain the edge x_i. Therefore, $f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) = f(N^m_{\mathcal{H}}[x_k]) - k(x_k)$. If $k = i_2$, then $f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) = f'(N^m_{\mathcal{H}}[x_{i_2}]) - k(x_{i_2})$ and $N^m_{\mathcal{H}}[x_k]$ contains the edge x_i. By Lemma 1, $f'(N^m_{\mathcal{H}}[x_{i_2}]) - k(x_{i_2}) = v_{f'}(x_{i_2}) + e_{f'}(x_{i_2}) - k(x_{i_2})$. By Statements (1) and (2) of this lemma, we know that $v_{f'}(x_{i_2}) + e_{f'}(x_{i_2}) - k(x_j) = v_f(x_{i_2}) + e_{f'}(x_{i_2}) - k(x_j) \le v_f(x_{i_2}) + e_f(x_{i_2}) - k(x_{i_2})$. Therefore, $n_{f'}(x_j) = \min\{n_f(x_j), v_f(x_{i_2}) + e_{f'}(x_{i_2}) - k(x_{i_2})\}$.

We now consider an edge x_k in $E_{\mathcal{H}}(x_j)$ with $k < j$. Recall that $x_t = (x_{i_2}, x_j)$. Clearly, $t < j$. If $k \neq t$, then $N^m_{\mathcal{H}}[x_k]$ does not contain the edge x_i. Therefore, $f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) = f(N^m_{\mathcal{H}}[x_k]) - k(x_k)$. If $k = t$, then $f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) = f'(N^m_{\mathcal{H}}[x_t]) - k(x_t)$ and $N^m_{\mathcal{H}}[x_k]$ contains the edge x_i. Note that $f'(x_i) = \mathcal{B}$, and $f'(x) = f(x)$ for every element $x \in V_{\mathcal{H}} \cup E_{\mathcal{H}} \setminus \{x_i\}$. By Lemma 1 and Statement (2) of this lemma, $f'(N^m_{\mathcal{H}}[x_t]) - k(x_t) = e_{f'}(x_{i_2}) + e_{f'}(x_j) - f'(x_t) + f'(x_{i_2}) + f'(x_j) - k(x_t) = e_{f'}(x_{i_2}) + e_f(x_j) - f(x_t) + f(x_{i_2}) + f(x_j) - k(x_t)$. Hence, $m_{f'}(x_j) = \min\{f'(N^m_{\mathcal{H}}[x_k]) - k(x_k) \mid x_k \in E_{\mathcal{H}}(x_j) \text{ and } k < j\} = \min\{m_f(x_j), f'(N^m_{\mathcal{H}}[x_t]) - k(x_t)\} = \min\{m_f(x_j), e_{f'}(x_{i_2}) + e_f(x_j) - f(x_t) + f(x_{i_2}) + f(x_j) - k(x_t)\}$. □

Theorem 4. *Assume that Algorithm IPMD runs and does not stop in Step 2. Let $Y = \{I_1, I_1 + d, \ldots, I_1 + (\ell-1)\cdot d\}$ and let $f : V_{\mathcal{H}} \cup E_{\mathcal{H}} \to Y$ be the function of $\mathcal{H}$ at the beginning of the i-th iteration of Steps 3–9. Let x_i be an element in $\mathcal{H}$ with $t(x_i) = F_r$. Both at the beginning and at the end of each iteration of Steps 3–9, the function f satisfies the following conditions for each vertex x_j in $\mathcal{H}$:*

(1) $v_f(x_j) = f(N_{\mathcal{H}}[x_j])$,

(2) $e_f(x_j) = f(E_{\mathcal{H}}(x_j))$,

(3) $n_f(x_j) = \min\{f(N^m_{\mathcal{H}}[x_k]) - k(x_k) \mid x_k \in N_{\mathcal{H}}[x_j] \text{ and } k \le j\}$, *and*

(4) $m_f(x_j) = \min\{f(N^m_{\mathcal{H}}[x_k]) - k(x_k) \mid x_k \in E_{\mathcal{H}}(x_j) \text{ and } k < j\}$.

Proof. Clearly, the function f at the beginning of the first iteration of Steps 3–9 satisfies the four conditions for each vertex x_j in $\mathcal{H}$. We assume that the function f at the beginning of the i-th iteration of Steps 3–9 satisfies the four conditions for each vertex x_j in $\mathcal{H}$. By Lemmas 4 and 5, we know that through Procedure UPDATE, the new function f obtained by changing the value of $f(x_i)$ in Step 6 also satisfies the four conditions for each vertex x_j in $\mathcal{H}$ at the end of the i-th iteration of Steps 3–9. Note that the function f at the end of the i-th iteration of Steps 3–9 is the function f at the beginning of the $(i+1)$-th iteration. Hence, the theorem holds. □

The following theorem can be easily verified.

Theorem 5. *For each element x_i in $\mathcal{H}$ with $t(x_i) = F_r$, the running time of Procedure* UPDATE *is $O(deg_{\mathcal{H}}(x_i))$ if $x_i \in V_{\mathcal{H}}$, and $O(1)$ if $x_i \in E_{\mathcal{H}}$.*

Theorem 6. *Algorithm IPMD finds a $\mathcal{P}$-mixed dominating function of a rooted tree $\mathcal{H}$ of minimum weight in $O(n)$ time.*

Proof. By Theorems 1–4, Algorithm IPMD is equivalent to Algorithm PMD and thus the function f returned from Step 10 of Algorithm IPMD is a $\mathcal{P}$-mixed dominating function of $\mathcal{H}$ of minimum weight.

Note that $|V_{\mathcal{H}}| = n$ and $|E_{\mathcal{H}}| = m = n - 1$. By Theorems 1–3 and Theorem 5, the running time of Algorithm IPMD is

$$O\left(\sum_{x_i \in V_{\mathcal{H}}} (deg_{\mathcal{H}}(x_i) + 1)\right) = O(2m + n) = O(n).$$

□

References

[1] Y. Alavi, M. Behzad, L.M. Lesniak-Foster, E.A. Nordhaus, Total matchings and total coverings of graphs, Journal of Graph Theory 1 (1977), 135–140.

[2] Y. Alavi, , J. Liu, J. Wang, Z. Zhang, On total covers of graphs, Discrete Mathematics 100 (1992), 229–233.

[3] J. Aleš, R. Bačík, Strong elimination ordering of the total graph of a tree, Discrete Applied Mathematics 39 (1992), 293–295.

[4] G.S. Adhar, S. Peng, Mixed domination in tree: a parallel algorithm, in: Proceedings of the 25th Southeastern International Conference on Combinatorics, Graph Theory, and Computing, Congr. Numer. 100 (1994), 73–80.

[5] G.J. Chang, Algorithmic aspects of domination in graphs, in: Du, D.Z, Pardalos, P.M. (eds.) Handbook of Combinatorial Optimization, Vol. 3, pp. 339–405, Kluwer, Boston, MA, 1998.

[6] Y.H. Chen, The mixed dominating set problem is MAX SNP-hard, in: Proceedings of the 29th Workshop on Combinatorial Mathematics and Computation Theory, 120–123, 2012.

[7] P. Erdös, A. Meir, On total matching numbers and total covering numbers of complementary graphs, Discrete Mathematics 187 (1998), 269–271.

[8] M. Farber, Characterizations of strongly chordal graphs, Discrete Mathematics 43 (1983), 173–189.

[9] P. Hatami, An approximation algorithm for the total cover problem, Discussiones Mathematicae Graph Theory 27 (2007), 553–560.

[10] T.W. Haynes, S.T. Hedetniemi, P.J. Slater, Fundamentals of Domination in Graphs, Marcel Dekker, New York, 1998.

[11] T.W. Haynes, S.T. Hedetniemi, P.J. Slater, Domination in Graphs: Advanced Topics, Marcel Dekker, New York, 1998.

[12] J.K. Lan, G.J. Chang, , On the mixed domination problem in graphs, Theoretical Computer Science 476 (2013), 84–93.

[13] C.-M. Lee, M.-S. Chang, Variations of Y-dominating functions on graphs, Discrete Mathematics 308 (2008), 4185–4204.

[14] Y.T. Li, J.J. Liu, Y.L. Wang, On total covers of block-cactus graphs, in: Proceedings of Advances in Intelligent Systems and Applications, SIST20, 33–39, 2013.

[15] D.F. Manlove, On the algorithmic complexity of twelve covering and independence parameters of graphs, Discrtete Applied Mathematics 91 (1999), 155–175.

[16] R. Paige, R.E. Tarjan, Three partition refinement algorithms. SIAM Journal on Computing 16 (1987), 973–989.

[17] D.J. Rose, Triangulated graphs and the elimination process, Journal of Mathematical Analysis and Applications 32 (1970), 597–609.

[18] P.J. Slater, Domination and location in acyclic graphs, Networks 17 (1987), 55–64.

[19] J.P. Spinrad, Doubly lexical ordering of dense 0-1 matrices, Information Processing Letters 45(1993), 229–235.

[20] Y. Zhao, L. Kang, M.Y. Sohn, The algorithmic complexity of mixed domination in graphs, Theoretical Computer Science 412 (2001), 2387–2392.