Opuscula Math. 45, no. 6 (2025), 841–855
https://doi.org/10.7494/OpMath.2025.45.6.841 Opuscula Mathematica

A LINEAR TIME ALGORITHM TO COMPUTE VERTICES THAT BELONG TO ALL, SOME AND NO MINIMUM DOMINATING SETS IN A TREE AND ITS CONSEQUENCES

Radosław Ziemann and Paweł Żyliński

Communicated by Andrzej Żak

Abstract. We provide a linear time algorithm for determining the sets of vertices that belong to all, some and no minimum dominating sets of a tree, respectively, thus improving the quadratic time algorithm of Benecke and Mynhardt in 2008 [S. Benecke, C.M. Mynhardt, Trees with domination subdivision number one, Australas. J. Comb. 42 (2008), 201–209]. Some algorithmic consequences are also discussed.

Keywords: tree, domination, independence, vertex cover, dissociation, algorithm, linear time, dynamic programming.

Mathematics Subject Classification: 05C05, 05C69, 05C85.

1. INTRODUCTION

The reader is referred to [4, 35] for definitions and notations in graph theory. A subset D of V_G is said to be a *dominating set* of a graph $G = (V_G, E_G)$ if each vertex belonging to the set $V_G \setminus D$ has a neighbour in D. A set $X \subseteq V_G$ is called an *independent set* in G if any pairwise distinct vertices in X are not adjacent. If a set $D \subseteq V_G$ is both a dominating set and an independent set, then it is an *independent dominating set* of G. The (*independent*) *domination number* of G, denoted by $\gamma(G)$ (resp., by $\gamma_i(G)$), is defined to be the minimum cardinality of a (independent) dominating set D of G, and any minimum (independent) dominating set of G is referred to as a γ-set (resp., as a γ_i-set).

Next, we define the sets $\mathcal{A}_\gamma(G), \mathcal{S}_\gamma(G)$ and $\mathcal{N}_\gamma(G)$, respectively, by setting

$$\mathcal{A}_\gamma(G) = \{v \in V_G : v \text{ is in every } \gamma(G)\text{-set}\},$$
$$\mathcal{S}_\gamma(G) = \{v \in V_G : v \text{ is in some but not every } \gamma(G)\text{-set}\},$$

and

$$\mathcal{N}_\gamma(G) = \{v \in V_G : v \text{ is in no } \gamma(G)\text{-set}\}.$$

The concept of sets $\mathcal{A}_\gamma(\cdot)$ and $\mathcal{N}_\gamma(\cdot)$ was introduced by Mynhard in [33], see also [9], who characterized these sets for trees. Since then the analogous sets has been defined and studied for several graph domination parameters, see for example [5, 6, 14, 15, 25, 26, 32, 39], to mention just a few.

In this paper, the set $\mathcal{N}_\gamma(\cdot)$ is of our particular interest (although we do not lose sight of the set $\mathcal{S}_\gamma(\cdot)$ – since $\mathcal{S}_\gamma(T) = \emptyset$ implies the uniqueness of a γ-set – as well as of the set $\mathcal{A}_\gamma(\cdot)$ since $\mathcal{A}_\gamma(T) = \emptyset$ is a necessary condition for a tree T to satisfy the m-eternal eviction number $e_{\mathrm{m}}^{\infty}(T) = \gamma(T)$, see [27]). The reason is that all the trees with the set $\mathcal{N}_\gamma(\cdot)$ being empty constitutes the class of γ-excellent trees [20] (equivalent to the class of critically dominated trees or dot-critical trees, see Theorem 3.8 in [11] for more details, in particular, for tree classes equivalent to the class of γ-excellent trees), which have been used to obtain a constructive characterization of trees with equal domination and secure domination numbers [34]. Next, the concept of sets $\mathcal{N}_\gamma(\cdot)$ has been also utilized in order to characterize the class of trees with $\mathrm{sd}_\gamma = 1$ [3] and the class of trees with $\mathrm{sd}_{\gamma_i} = 1$ [2]. Recall that for a given graph parameter μ, the μ-subdivision number sd_μ is defined to be the minimum number of edges that must be subdivided, where each edge may be subdivided at most once, to change μ.

Taking into account the aforementioned applications (and probably many more) of sets $\mathcal{N}_\gamma(\cdot)$, an algorithmic point of view is desirable. In 2008, Benecke and Mynhardt [3] (explicitly) considered the problem of determining the set $\mathcal{N}_\gamma(T)$ for a given tree T, and provided a quadratic time algorithm based upon the structural characterization of vertices that belongs to no minimum dominating set of a tree [33]. Next, in 2021, Bouquet *et al.* [9] established that the problem of verifying whether either $v \in \mathcal{A}_\gamma(G)$ or $v \in \mathcal{S}_\gamma(G)$, or $v \in \mathcal{N}_\gamma(G)$ can be solved in time $O(|V_G| + |E_G|)$ for any interval graph $G = (V_G, E_G)$, which immediately (although not discussed by the authors) resulted in a quadratic time algorithm for the problem of determining the sets $\mathcal{A}_\gamma(\cdot), \mathcal{S}_\gamma(\cdot)$ and $\mathcal{N}_\gamma(\cdot)$ in that class of graphs. To the best of our knowledge, these are the only two algorithmic results on the topic, and for a graph $G = (V_G, E_G)$ and a vertex $v \in V_G$, the complexity status of the problem of deciding whether either $v \in \mathcal{A}_\gamma(G)$ or $v \in \mathcal{S}_\gamma(G)$, or $v \in \mathcal{N}_\gamma(G)$ remains open in general. The only general (positive) result we are aware of is Property 2.7 in [9]: *Let $\mathcal{C}$ be a class of graphs such that the minimum dominating set problem is polynomial time solvable for the graphs G and $G_v + u$. Then, given a graph $G = (V_G, E_G) \in \mathcal{C}$ and vertex $v \in V_G$, the problem of deciding whether either $v \in \mathcal{A}_\gamma(G)$ or $v \in \mathcal{S}_\gamma(G)$, or $v \in \mathcal{N}_\gamma(G)$ is in P.* The aforementioned class $\mathcal{C}$ includes trees and interval graphs [9].

It should be emphasized that the standard dynamic programming-based algorithm for determining the domination number, described in [13], also allows to determine this set in quadratic time. Herein, based upon this algorithm, using the concept of re-rooting (see the paragraph below), we provide a linear time algorithm for determining the sets $\mathcal{A}_\gamma(T), \mathcal{S}_\gamma(T)$ and $\mathcal{N}_\gamma(T)$ for a given tree T.

Theorem 1.1. *For any tree T, the sets $\mathcal{A}_\gamma(T), \mathcal{S}_\gamma(T)$ and $\mathcal{N}_\gamma(T)$ can be determined in linear time and space.*

Our result immediately implies the following corollary.

Corollary 1.2. *For any tree T, the following problems are solvable in linear time and space.*

(a) *The problem of verifying whether T is γ-excellent.*
(b) *The problem of verifying whether $sd_{\gamma}(T) = 1$.*
(c) *The problem of verifying whether $sd_{\gamma_i}(T) = 1$.*

As already observed, if $\mathcal{S}_{\gamma}(T) = \emptyset$ then a tree T has a unique minimum dominating set. So our result also allows to identify the class of trees with unique dominating sets in linear time. However, as pointed by Gunther *et al.* [23], this problem can be solved efficiently much simpler: just by computing any γ-set D of a tree in linear time [13], and then checking whether each element in D has at least two external private neighbours (which can be done in linear time either).

The re-rooting technique is a common method exploited to efficiently compute values for every possible root r of a (rooted) tree T. It involves two main steps: first, computing a value for an initial arbitrary root, and second, using that information to efficiently calculate the values for all other nodes by effectively "re-rooting" the tree. To the best of our knowledge, there is no formal general description of this approach except several specific examples in some coding tutorials/competitions, see for example [42–44]. Therefore, we finalize our paper with slightly generalizing and briefly formalizing the method as well as illustrating it with three examples – in particular, we improve the quadratic algorithmic result in [40].

2. LINEAR TIME ALGORITHM

Let $G = (V_G, E_G)$ be a graph and consider a vertex $v \in V_G$. Define

$$\gamma^1(G, v) = \min\{|D| : D \text{ is a dominating set of } G \text{ and } v \in D\}$$

and

$$\gamma^0(G, v) = \min\{|D| : D \text{ is a dominating set of } G \text{ and } v \notin D\}.$$

One can observe that for any graph $G = (V_G, E_G)$, we have

$$\gamma(G) = \min(\gamma^1(G, v), \gamma^0(G, v))$$

for any vertex $v \in V_G$. Next, define

$$\gamma^{00}(G, v) = \min\{|D| : D \text{ is a dominating set of } G - \{v\}\}.$$

(Recall that for a graph $G = (V_G, E_G)$ and a vertex subset $X \subseteq V_G$, $G - X$ denotes the graph resulting from G by deleting all the vertices in X, together with all their incident edges.) Note that $\gamma^{00}(G, v) \leq \gamma^0(G, v)$, since a dominating set D of G with $v \notin D$ is also a dominating set of $G - v$. We have the following theorem (see Figure 1 for an illustration).

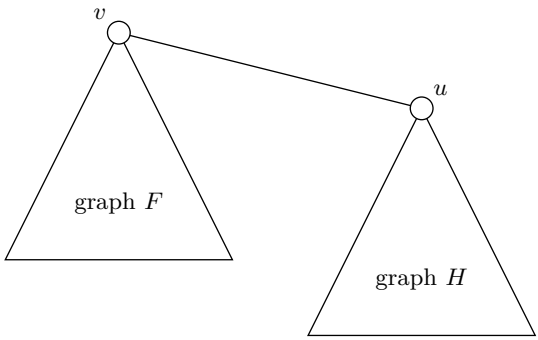

Fig. 1. Illustration for Theorem 2.1

Theorem 2.1 ([13]). *Suppose $F = (V_F, E_F)$ and $H = (V_H, E_H)$ are graphs with specified vertices $v \in V_F$ and $u \in V_H$, respectively ($V_F \cap V_H = \emptyset$). Let $G = (V_G, E_G)$ be the graph with the specified vertex v, which is obtained from the disjoint union of F and H by adding the new edge $\{v, u\}$. Then the following statements hold:*

(1) $\gamma^1(G, v) = \gamma^1(F, v) + \min\{\gamma^1(H, u), \gamma^{00}(H, u)\}$,
(2) $\gamma^0(G, v) = \min\{\gamma^0(F, v) + \gamma^0(H, u), \gamma^{00}(F, v) + \gamma^1(H, u)\}$,
(3) $\gamma^{00}(G, v) = \gamma^{00}(F, v) + \gamma(H) = \gamma^{00}(F, v) + \min\{\gamma^1(H, u), \gamma^0(H, u)\}$.

Theorem 2.1 gives rise to a simple – based upon dynamic programming method [17] – algorithm for the problem of determining the domination number of a tree T [13]: root the input tree $T = (V_T, E_T)$ at any vertex $v \in V_T$ and, in a bottom-up fashion, compute the relevant values γ^1, γ^0 and γ^{00} for all the roots of all relevant rooted subtrees, starting with the base case $\gamma^1(l) = 1, \gamma^0(l) = \infty$ and $\gamma^{00}(l) = 0$ for any leaf l of T; in a moment, we shall exploit this idea in detail.

Observe now that for any vertex $v \in V_T$ of a tree $T = (V_T, E_T)$, we have:

(i) $v \in \mathcal{A}_\gamma(T)$ if and only if $\gamma(T) \neq \gamma^0(T, v)$,
(ii) $v \in \mathcal{S}_\gamma(T)$ if and only if $\gamma^1(T, v) = \gamma^0(T, v)$,
(iii) $v \in \mathcal{N}_\gamma(T)$ if and only if $\gamma(T) \neq \gamma^1(T, v)$,

and therefore, we can simply determine the sets $\mathcal{A}_\gamma(T), \mathcal{S}_\gamma(T)$ and $\mathcal{N}_\gamma(T)$ in quadratic time, using the aforementioned dynamic programming-based algorithm. However, by exchanging the roles of graphs F and H in Theorem 2.1, one can observe that there is some local correlation between the relevant values of $\gamma^1(T, v)$ and $\gamma^1(T, u)$, and $\gamma^0(T, v)$ and $\gamma^0(T, u)$, respectively, for any two adjacent vertices in T, which immediately allows us to improve the running time up to linear time, which the next subsection is devoted to.

2.1. PRE-PROCESSING

Let $T = (V_T, E_T)$ be a tree and consider a vertex $r \in V_T$. Let T_r denote the tree T rooted at r, let $C_r(v)$ denote the (arbitrarily) ordered set of all children of a vertex

$v \in V_T$, and let $v_i \in C_r(v)$ denote the i-th child of v for $i \in \{1, \ldots, c_r(v)\}$, where $c_r(v)$ denotes the cardinality of $C_r(v)$. Next, let $T_{r,v}$ denote the subtree in T_r, rooted at v, containing v and all its descendants in T_r (and so $T_{r,r} = T_r$). Now – see Figure 2 for an illustration – if v is non-leaf vertex in T, we define the following *partial* subtrees for $i \in \{1, \ldots, c_r(v)\}$:

$$T_{r,v}^{\triangleright i} = T_{r,v} - \bigcup_{j=i}^{c_r(v)} V_{T_{r,v_j}} \text{ (called the } i\text{-prefix subtree)},$$

$$T_{r,v}^{i\triangleleft} = T_{r,v} - \bigcup_{j=1}^{i} V_{T_{r,v_j}} \text{ (called the } i\text{-suffix subtree)},$$

$$T_{r,v}^{i} = T_{r,v} - V_{T_{r,v_i}} \text{ (called the } i\text{-orphaned tree)}.$$

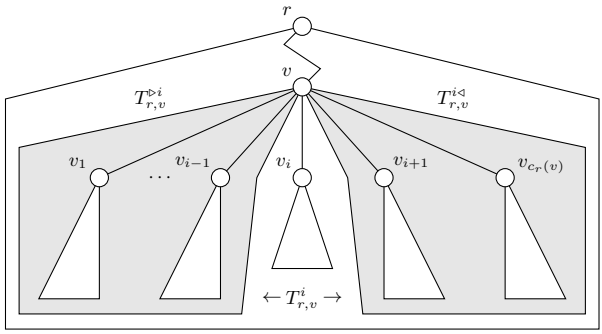

Fig. 2. Trees $T_{r,v}^{\triangleright i}$, $T_{r,v}^{i\triangleleft}$ and $T_{r,v}^{i}$

It follows from Theorem 2.1 that we can recursively compute all the values γ^1, γ^0 and γ^{00} for all the aforementioned subtrees: the prefix, suffix and orphaned ones. Namely, if v is a leaf, then – following [13] – we set

$$\gamma^1(T_{r,v}, v) = 1, \quad \gamma^0(T_{r,v}, v) = \infty \quad \text{and} \quad \gamma^{00}(T_{r,v}, v) = 1$$

(notice that in this case, no partial subtree exists). Otherwise, if v is a non-leaf vertex, assuming that all the relevant values have been computed for all v's children in $C_r(v)$, then the following formulas apply.

Tree $T_{r,v}^{\triangleright 1}$ – the base case ($i = 1$). We set (by the definition):

$$\gamma^1(T_{r,v}^{\triangleright 1}, v) = 1, \quad \gamma^0(T_{r,v}^{\triangleright 1}, v) = \infty, \quad \gamma^{00}(T_{r,v}^{\triangleright 1}, v) = 0.$$

Trees $T^{\triangleright i}_{r,v}$ – recursion ($i > 1$). We set (by Theorem 2.1):

$$\begin{aligned}
\gamma^1(T^{\triangleright i}_{r,v}, v) &= \gamma^1(T^{\triangleright i-1}_{r,v}, v) + \min\{\gamma^1(T_{r,v_{i-1}}, v_{i-1}), \gamma^{00}(T_{r,v_{i-1}}, v_{i-1})\},\\
\gamma^0(T^{\triangleright i}_{r,v}, v) &= \min\{\gamma^0(T^{\triangleright i-1}_{r,v}, v) + \gamma^0(T_{r,v_{i-1}}, v_{i-1}), \gamma^{00}(T^{\triangleright i-1}_{r,v}, v) + \gamma^1(T_{r,v_{i-1}}, v_{i-1})\},\\
\gamma^{00}(T^{\triangleright i}_{r,v}, v) &= \gamma^{00}(T^{\triangleright i-1}_{r,v}, v) + \min\{\gamma^1(T_{r,v_{i-1}}, v_{i-1}), \gamma^0(T_{r,v_{i-1}}, v_{i-1})\}.
\end{aligned}$$

Tree $T^{c_r(v)\triangleleft}_{r,v}$ – the base case ($i = c_r(v)$). We set (by the definition):

$$\gamma^1(T^{c_r(v)\triangleleft}_{r,v}, v) = 1, \quad \gamma^0(T^{c_r(v)\triangleleft}_{r,v}, v) = \infty, \quad \gamma^{00}(T^{c_r(v)\triangleleft}_{r,v}, v) = 0.$$

Trees $T^{i\triangleleft}_{r,v}$ – recursion. We set (by Theorem 2.1):

$$\begin{aligned}
\gamma^1(T^{i\triangleleft}_{r,v}, v) &= \gamma^1(T^{i+1\triangleleft}_{r,v}, v) + \min\{\gamma^1(T_{r,v_{i+1}}, v_{i+1}), \gamma^{00}(T_{r,v_{i+1}}, v_{i+1})\},\\
\gamma^0(T^{i\triangleleft}_{r,v}, v) &= \min\{\gamma^0(T^{i+1\triangleleft}_{r,v}, v) + \gamma^0(T_{r,v_{i+1}}, v_{i+1}), \gamma^{00}(T^{i+1\triangleleft}_{r,v}, v) + \gamma^1(T_{r,v_{i+1}}, v_{i+1})\},\\
\gamma^{00}(T^{i\triangleleft}_{r,v}, v) &= \gamma^{00}(T^{i+1\triangleleft}_{r,v}, v) + \min\{\gamma^1(T_{r,v_{i+1}}, v_{i+1}), \gamma^0(T_{r,v_{i+1}}, v_{i+1})\}.
\end{aligned}$$

Trees $T^1_{r,v}$ and $T^{c_r(v)}_{r,v}$ – the base case ($i = 1$ or $i = c_r(v)$). We set (by the definition):

$$\begin{aligned}
\gamma^1(T^1_{r,v}, v) &= \gamma^1(T^{1\triangleleft}_{r,v}, v),\\
\gamma^1(T^{c_r(v)}_{r,v}, v) &= \gamma^1(T^{\triangleright c_r(v)}_{r,v}, v),\\
\gamma^0(T^1_{r,v}, v) &= \gamma^0(T^{1\triangleleft}_{r,v}, v),\\
\gamma^0(T^{c_r(v)}_{r,v}, v) &= \gamma^0(T^{\triangleright c_r(v)}_{r,v}, v),\\
\gamma^{00}(T^1_{r,v}, v) &= \gamma^{00}(T^{1\triangleleft}_{r,v}, v),\\
\gamma^{00}(T^{c_r(v)}_{r,v}, v) &= \gamma^{00}(T^{\triangleright c_r(v)}_{r,v}, v).
\end{aligned}$$

Trees $T^i_{r,v}$ – recursion ($i \notin \{1, c_r(v)\}$). We set:

(1) $\gamma^1(T^i_{r,v}, v) = \gamma^1(T^{\triangleright i}_{r,v}, v) + \gamma^1(T^{i\triangleleft}_{r,v}, v) - 1.$
(2) $\gamma^0(T^i_{r,v}, v) = \min\{\gamma^0(T^{\triangleright i}_{r,v}, v) + \gamma^{00}(T^{i\triangleleft}_{r,v}, v), \gamma^{00}(T^{\triangleright i}_{r,v}, v) + \gamma^0(T^{i\triangleleft}_{r,v}, v)\}.$
(3) $\gamma^{00}(T^i_{r,v}, v) = \sum_{j=1, j\neq i}^{c_r(v)} \min\{\gamma^1(T_{r,v_j}, v_j), \gamma^0(T_{r,v_j}, v_j)\}.$

Formulas (1)–(3) require formal (elementary) proofs – for clarity of presentation, they are postponed to Section 3.

Tree T_v – recursion (v is not a leaf). We set (by Theorem 2.1):

$$\begin{aligned}
\gamma^1(T_{r,v}, v) &= \gamma^1(T^{\triangleright c_r(v)}_{r,v}, v) + \min\{\gamma^1(T_{r,v_{c_r(v)}}, v_{c_r(v)}), \gamma^{00}(T_{r,v_{c_r(v)}}, v_{c_r(v)})\},\\
\gamma^0(T_v, v) &= \min\{\gamma^0(T^{\triangleright c_r(v)}_{r,v}, v) + \gamma^0(T_{r,v_{c_r(v)}}, v_{c_r(v)}),\\
&\qquad \gamma^{00}(T^{\triangleright c_r(v)}_{r,v}, v) + \gamma^1(T_{r,v_{c_r(v)}}, v_{c_r(v)})\},\\
\gamma^{00}(T_v, v) &= \gamma^{00}(T^{\triangleright c_r(v)}_{r,v}, v) + \min\{\gamma^1(T_{r,v_{c_r(v)}}, v_{c_r(v)}), \gamma^0(T_{r,v_{c_r(v)}}, v_{c_r(v)})\}.
\end{aligned}$$

As regards computing time, it follows immediately from the construction that all but the value described by Formula (3) can be computed in time $O(\deg(v))$, assuming that all relevant data for v's children in $C_r(v)$ has been already computed and stored. Consider now Formula (3). It follows from the definition that

$$\begin{aligned}\gamma^{00}(T^{i+1}_{r,v}, v) = \gamma^{00}(T^{i}_{r,v}, v) \\ - \min\{\gamma^1(T_{r,v_{i+1}}, v_{i+1}), \gamma^0(T_{r,v_{i+1}}, v_{i+1})\} \\ + \min\{\gamma^1(T_{r,v_i}, v_i), \gamma^0(T_{r,v_i}, v_i)\},\end{aligned}$$

which allows us to compute all $\gamma^{00}(T^i_{r,v}, v)$'s in total time $O(\deg(v))$.

Consequently, given an n-vertex tree $T = (V_T, E_T)$ and a vertex $r \in V_T$, all the relevant data in the rooted tree T_r, for each of its vertices, can be computed in total $\sum_{v \in V_T} O(\deg(v)) = O(n)$ time and space, using a bottom-up approach (again see [13]); in the following, we shall refer to this step as *pre-processing step*.

2.2. COMPUTING THE SETS $\mathcal{A}_\gamma(\cdot)$, $\mathcal{S}_\gamma(\cdot)$ AND $\mathcal{N}_\gamma(\cdot)$

Let r be a fixed vertex of a non-trivial n-vertex tree $T = (V_T, E_T)$, assume that the pre-processing step for the rooted tree T_r has already been done, and recall equivalence $T_u = T_{u,u}$ in notion for any $u \in V_T$. Consider now a vertex $x \in V_T$ (initially, we take $x = r$), with non-empty set $C_r(x)$, such that the values $\gamma^1(T, x), \gamma^0(T, x)$ and $\gamma^{00}(T, x)$ as well as all the values $\gamma^1(T^i_{x,x}, x)$, $\gamma^0(T^i_{x,x}, x)$ and $\gamma^{00}(T^i_{x,x}, x)$, $i = 1, \ldots, c_r(x)$, have been already determined. It follows from Theorem 2.1 that (see Figure 3 for an illustration):

$$\begin{aligned}\gamma^1(T, x_i) &= \gamma^1(T_{r,x_i}, x_i) + \min\{\gamma^1(T^i_{x,x}, x), \gamma^{00}(T^i_{x,x}, x)\}, \\ \gamma^0(T, x_i) &= \min\{\gamma^0(T_{r,x_i}, x_i) + \gamma^0(T^i_{x,x}, x), \gamma^{00}(T_{r,x_i}, x_i) + \gamma^1(T^i_{x,x}, x)\}, \\ \gamma^{00}(T, x_i) &= \gamma^{00}(T_{r,x_i}, x_i) + \min\{\gamma^1(T^i_{x,x}, x), \gamma^0(T^i_{x,x}, x)\}.\end{aligned}$$

Furthermore, if $C_r(x_i)$ is non-empty, then, for any $j \in \{1, \ldots, c_r(x_i)\}$ we obtain:

$$\begin{aligned}\gamma^1(T^j_{x_i,x_i}, x_i) &= \gamma^1(T^j_{r,x_i}, x_i) + \min\{\gamma^1(T^i_{x,x}, x), \gamma^{00}(T^i_{x,x}, x)\}, \\ \gamma^0(T^j_{x_i,x_i}, x_i) &= \min\{\gamma^0(T^j_{r,x_i}, x_i) + \gamma^0(T^i_{x,x}, x), \gamma^{00}(T^j_{r,x_i}, x_i) + \gamma^1(T^i_{x,x}, x)\}, \\ \gamma^{00}(T^j_{x_i,x_i}, x_i) &= \gamma^{00}(T^j_{r,x_i}, x_i) + \min\{\gamma^1(T^i_{x,x}, x), \gamma^0(T^i_{x,x}, x)\}.\end{aligned}$$

Clearly, after pre-processing step (done only once for the rooted tree T_r), all the aforementioned data can be computed in $O(\deg(x_i))$ time for each vertex $x_i \in C_r(x)$. Consequently, combining the above construction with a BFS-like approach, we can compute $\gamma^1(T, v)$ and $\gamma^0(T, v)$ for each vertex $v \in V_T$ in total $O(n)$ time and space, and so identify all vertices in the sets $\mathcal{A}_\gamma(T), \mathcal{S}_\gamma(T)$ and $\mathcal{N}_\gamma(T)$, respectively, which completes the proof of Theorem 1.1.

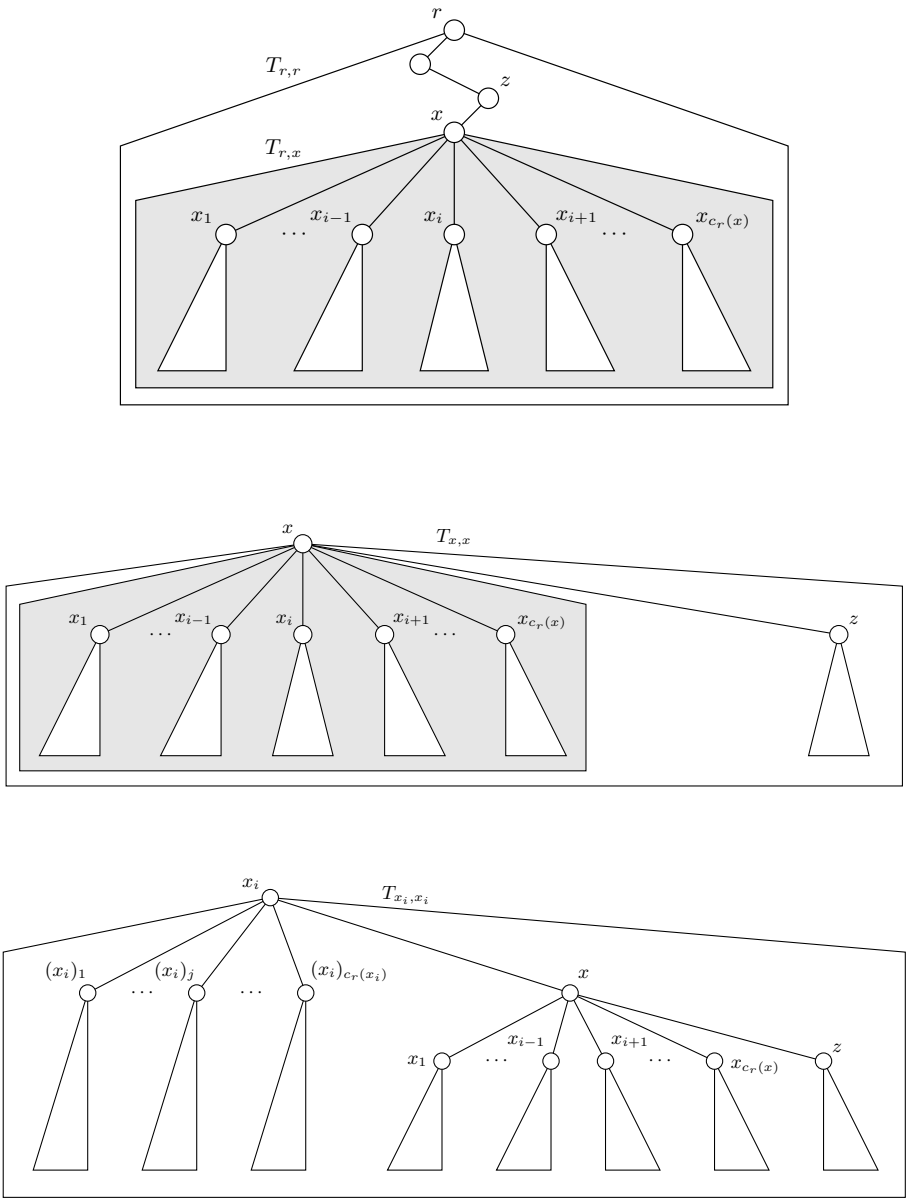

Fig. 3. Computing $\mathcal{N}_\gamma(T)$ using a BFS-like approach

3. PROOFS OF FORMULAS (1)–(3)

Let us recall Formulas (1)–(3) to be proved.

(1) $\gamma^1(T^i_{r,v}, v) = \gamma^1(T^{\triangleright i}_{r,v}, v) + \gamma^1(T^{i\triangleleft}_{r,v}, v) - 1$.
(2) $\gamma^0(T^i_{r,v}, v) = \min\{\gamma^0(T^{\triangleright i}_{r,v}, v) + \gamma^{00}(T^{i\triangleleft}_{r,v}, v), \gamma^{00}(T^{\triangleright i}_{r,v}, v) + \gamma^0(T^{i\triangleleft}_{r,v}, v)\}$.
(3) $\gamma^{00}(T^i_{r,v}, v) = \sum_{j=1, j\neq i}^{c_r(v)} \min\{\gamma^1(T_{r,v_j}, v_j), \gamma^0(T_{r,v_j}, v_j)\}$.

In order to prove Formulas (1)–(3), all we need is to establish a counterpart of Theorem 2.1 in case of identifying vertices v and u. Namely, we have the following theorem (see Figure 4 for an illustration).

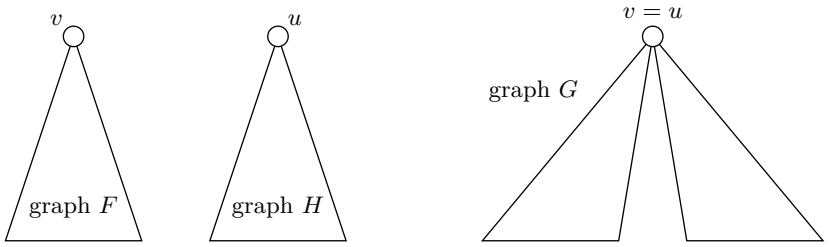

Fig. 4. Illustration for Theorem 3.1

Theorem 3.1. *Let $F = (V_F, E_F)$ and $H = (V_H, E_H)$ be two vertex disjoint graphs with specified vertices $v \in V_F$ and $u \in V_H$, respectively. Let $G = (V_G, E_G)$ be the graph with the specified vertex v, which is obtained from F and H by identifying vertex u with v. Then:*

(1) $\gamma^1(G, v) = \gamma^1(F, v) + \gamma^1(H, u) - 1$.
(2) $\gamma^0(G, v) = \min\{\gamma^0(F, v) + \gamma^{00}(H, u), \gamma^{00}(F, v) + \gamma^0(H, u)\}$.
(3) $\gamma^{00}(G, v) = \gamma^{00}(F, v) + \gamma^{00}(H, u)$.

Proof. Formulas (1) and (3) follow straightforward from the definitions of domination, the two parameters $\gamma^1(\cdot)$ and $\gamma^{00}(\cdot)$, and the graph G. As regards Formula (2), we first observe that

$$\begin{aligned}\gamma^0(G, v) &\leq \min\{\gamma^0(F, v) + \min\{\gamma^0(H, u), \gamma^{00}(H, u)\},\\ &\qquad \min\{\gamma^0(F, v), \gamma^{00}(F, v)\} + \gamma^0(H, u)\}\\ &= \min\{\gamma^0(F, v) + \gamma^{00}(H, u), \gamma^{00}(F, v) + \gamma^0(H, u)\},\end{aligned}$$

since we have $\gamma^{00}(F, v) \leq \gamma^0(F, v)$ and $\gamma^{00}(H, u) \leq \gamma^0(H, u)$. On the other hand, let D_0 be a minimum dominating set of G such that $v \notin D_0$. Let $D_0^F = D_0 \cap V_F$ and $D_0^H = D_0 \cap V_H$. We can distinguish two cases.

Case 1. Vertex v is dominated by a vertex $v_F \in V_F$. Then, we obtain

$$\gamma^0(G) = |D_0^F| + |D_0^H| \geq \gamma^0(F, v) + \min\{\gamma^0(H, u), \gamma^{00}(H, u)\}.$$

Case 2. Vertex v is dominated by a vertex $v_H \in V_H$. Then, we obtain

$$\gamma^0(G) = |D_0^F| + |D_0^H| \geq \min\{\gamma^0(F,v), \gamma^{00}(F,v)\} + \gamma^0(H,u).$$

Therefore,

$$\gamma^0(G,v) \geq \min\{\gamma^0(F,v) + \min\{\gamma^0(H,u), \gamma^{00}(H,u)\}, \\ \min\{\gamma^0(F,v), \gamma^{00}(F,v)\} + \gamma^0(H,u)\}$$

as required, which finishes the proof. □

4. GENERAL APPROACH

For a given n-vertex graph $G = (V_G, E_G)$, let Π be some vertex property. We are interested in determining the subset $\Pi(G)$ of all vertices in V_G that satisfy Property Π in G.

Assume now that the (TRUE/FALSE)-property "$v \in \Pi(G)$" is expressible as a function $f_\Pi(\Pi^1(G,v), \ldots, \Pi^c(G,v))$ of some c vertex parameters $\Pi^1(G,v), \ldots, \Pi^c(G,v)$ in a given graph $G = (V_G, E_G)$. Next, assume that one can prove the following two properties.

1. Let $F = (V_F, E_F)$ and $H = (V_H, E_H)$ be two vertex disjoint graphs with specified vertices $v \in V_F$ and $u \in V_H$, respectively. Let $G = (V_G, E_G)$ be the graph with the specified vertex v, which is obtained from the disjoint union of F and H by adding the new edge $\{v,u\}$. Then for each $i = 1, \ldots, c$, we have:

$$\Pi^i(G,v) = f^i(\Pi^1(F,v), \Pi^1(H,u), \ldots, \Pi^c(F,v), \Pi^c(H,u)),$$

   where each $f^i(\cdot)$ is a function of $2c$ vertex parameters.
2. Let $F = (V_F, E_F)$ and $H = (V_H, E_H)$ be two vertex disjoint graphs with specified vertices $v \in V_F$ and $u \in V_H$, respectively. Let $G = (V_G, E_G)$ be the graph with the specified vertex v, which is obtained from F and H by identifying vertex u with v. Then for each $i = 1, \ldots, c$, we have

$$\Pi^i(G,v) = g^i(\Pi^1(F,v), \Pi^1(H,u), \ldots, \Pi^c(F,v), \Pi^c(H,u)),$$

   where each $g^i(\cdot)$ is a function of $2c$ vertex parameters.

Finally, assume that each of the functions $f, f^1, \cdots, f^c, g^1, \cdots, g^c$ is computable in time $O(c)$. Then, based upon Properties 1–2, by exploiting dynamic programming approach enriched with the concept of the rooted prefix, suffix and orphaned subtree (the relevant subtrees $T_{r,v}^{\triangleright i}, T_{r,v}^{\triangleleft i}$ and $T_{r,v}^i$ in Section 2, respectively), one can conclude that for any tree $T = (V_T, E_T)$, the problem of determining the set $\Pi(T)$ can be solved in time

$$\sum_{u \in v_T} O(\deg_T(u)) \cdot O(c) = O(n \cdot c),$$

which, if c is a constant number, immediately results in time $O(n)$ as required. In below, we briefly discuss three exemplary problems which this method can be successfully applied to.

Maximum independent set. Recall that a set $X \subseteq V_G$ is called an *independent set* in a graph $G = (V_G, E_G)$ if any pairwise distinct vertices in X are not adjacent; the size of a maximum independent set in G is denoted by $\alpha(G)$ and we refer to any maximum independent set of G as $\alpha(G)$-*set*. Define the sets $\mathcal{A}_\alpha(G)$, $\mathcal{S}_\alpha(G)$ and $\mathcal{N}_\alpha(G)$, respectively, by setting

$$\mathcal{A}_\alpha(G) = \{v \in V_G : v \text{ is in every } \alpha(G)\text{-set}\} \quad (\text{the } \textit{core} \text{ of } G),$$
$$\mathcal{S}_\alpha(G) = \{v \in V_G : v \text{ is in some but not every } \alpha(G)\text{-set}\} \quad (\text{the } \textit{corona} \text{ of } G),$$

and

$$\mathcal{N}_\alpha(G) = \{v \in V_G : v \text{ is in no } \alpha(G)\text{-set}\} \quad (\text{the } \textit{anti-core} \text{ of } G).$$

The idea of characterizing the sets $\mathcal{A}_\alpha(\cdot)$ and $\mathcal{N}_\alpha(\cdot)$ has been widely studied in the literature, see for example [8, 24, 30, 31]. For a given graph G, the problem of deciding whether there are vertices in G belonging to core(G) is NP-hard; on the other hand, it has been noticed that for hereditary families of graphs, where computing the independence number $\alpha(G)$ is polynomial, the problem can be solved in polynomial time either [8]. As regards the class of trees, the two general properties: (a) $v \in \mathcal{A}_\alpha(G)$ if and only if $\alpha(G-v) < \alpha(G)$ and (b) $v \in \mathcal{N}_\alpha(G)$ if and only if $\alpha(G-N_G[v]) = \alpha(G)-1$ for any graph G – see Lemma 4 and 6 in [8] – allows us to compute $\mathcal{A}_\alpha(T)$, $\mathcal{S}_\alpha(T)$ and $\mathcal{N}_\alpha(T)$ in linear time for any tree T, also by using the aforementioned re-rooting technique. However, here, due to a much simpler nature of the problem, there is no need to exploit the concept of prefix and suffix subtrees.

Minimum vertex cover. Recall that a set $X \subset V_G$ is called a *vertex cover* in a graph $G = (V_G, E_G)$ if every edge in E_G is incident to some element in X; the size of a minimum vertex cover in G is denoted by $\tau(G)$ and we refer to any maximum independent set of G as $\tau(G)$-*set*. Define the sets $\mathcal{A}_\tau(G)$, $\mathcal{S}_\tau(G)$ and $\mathcal{N}_\tau(G)$, respectively, by setting

$$\mathcal{A}_\tau(G) = \{v \in V_G : v \text{ is in every } \tau(G)\text{-set}\},$$
$$\mathcal{S}_\tau(G) = \{v \in V_G : v \text{ is in some but not every } \tau(G)\text{-set}\},$$

and

$$\mathcal{N}_\tau(G) = \{v \in V_G : v \text{ is in no } \tau(G)\text{-set}\}.$$

To the best of our knowledge, the idea of characterizing as well as computing the sets $\mathcal{A}_\tau(\cdot)$, $\mathcal{S}_\tau(\cdot)$ and $\mathcal{N}_\tau(\cdot)$ has not been studied in the literature. The aforementioned approach allows us to compute $\mathcal{A}_\tau(T)$, $\mathcal{S}_\tau(T)$ and $\mathcal{N}_\tau(T)$ in linear time for any tree T; here again, due to a much simpler nature of the problem, there is no need to exploit the concept of prefix and suffix subtrees.

Maximum dissociation set. A *dissociation set* in a graph $G = (V_G, E_G)$ is a subset of vertices $X \subseteq V_G$ such that the induced subgraph $G[X]$ has maximum degree at most 1. The *dissociation number* $\psi(G)$ of G – also known as the 1-*dependence number* [19] – is the cardinality of a maximum dissociation set of G. The concept of dissociation sets was introduced by Yannakakis [41] and is a natural generalization of independent set. The problem of finding a maximum dissociation set in a graph has been extensively studied on various sub-classes of graphs [1, 10, 12, 36] and is NP-hard for bipartite graphs [41].

Define the sets $\mathcal{A}_\psi(G)$, $\mathcal{S}_\psi(G)$ and $\mathcal{N}_\psi(G)$, respectively, by setting

$$\mathcal{A}_\psi(G) = \{v \in V_G : v \text{ is in every } \psi(G)\text{-set}\},$$
$$\mathcal{S}_\psi(G) = \{v \in V_G : v \text{ is in some but not every } \psi(G)\text{-set}\},$$

and

$$\mathcal{N}_\psi(G) = \{v \in V_G : v \text{ is in no } \psi(G)\text{-set}\}.$$

In 2022, Tu *et al.* [40] proposed a quadratic time algortihm for computing the sets $\mathcal{A}_\psi(T)$, $\mathcal{S}_\psi(T)$ and $\mathcal{N}_\psi(T)$ in a tree $T = (V_T, E_T)$. Simliarly as the authors in [3], they first propose the algortihm to identify whether a vertex $v \in V_T$ belongs to the set $\mathcal{A}_\psi(T)$ (resp. $\mathcal{N}_\psi(T)$), and then run that algorithm as a subroutine, which results in total quadratic time (recall $\mathcal{S}_\psi(T) = V_T - (\mathcal{A}_\psi(T) \cup \mathcal{N}_\psi(T))$). However, one can use the re-rooting technique, now supported with the concept of by prefix and suffix trees, to determine $\mathcal{A}_\psi(T)$, $\mathcal{S}_\psi(T)$ and $\mathcal{N}_\psi(T)$ in linear time.

5. CONCLUSIONS

We believe that the results presented here may be extended to tree-like graph classes, such as graphs of bounded treewidth [7] or bounded clique-width [16]. Nevertheless, the existing recursive formulations – unless ingeniously reformulated – likely become even more intricate. Therefore, we leave this direction as a worthwhile challenge.

Enumerating all minimum dominating sets. Clearly, one can design (a folk result) a simple algorithm for enumerating all minimum dominating sets directly based on the standard dynamic programming-based algorithm for determining the domination number [13] (the one we exploited in the paper). It is natural to inquire whether knowledge of the sets $\mathcal{A}_\gamma(\cdot)$, $\mathcal{S}_\gamma(\cdot)$ and $\mathcal{N}_\gamma(\cdot)$ may assist in enumerating all minimum dominating sets (see, for example, [18, 21, 37]). As already observed, if the set $\mathcal{S}_\gamma(\cdot)$ is empty, then the minimum dominating set is unique. What, then, happens in the opposite case? A complete answer to this question lies beyond the scope of the present study, and in what follows, we address only one specific aspect of this problem. (Recall that the problem of enumerating all *minimum* dominating sets in a tree differs substantially from that of enumerating all *minimal* dominating sets [22, 28, 38].)

Let $T = (V_T, E_T)$ be a tree and assume that we have already computed the sets $\mathcal{A}_\gamma(T)$, $\mathcal{F}_\gamma(T)$ and $\mathcal{N}_\gamma(T)$ in linear time. Consider now the forest $F = T[\mathcal{F}_\gamma(T)]$ induced by the (non-empty) set $\mathcal{F}_\gamma(T)$. Assume now that F consists of k stars $S_1 = (V_{S_1}, E_{S_1}), \ldots, S_k = (V_{S_k}, E_{S_k})$ (which is checkable in linear time). Following

the proof of Lemma 4 in [29], one can observe that for any γ-set D of T, we have $|D \cap V_{S_i}| = 1$, for each $i = 1, \ldots, k$. Consequently, keeping in mind the definition of the set $\mathcal{F}_\gamma(T)$, if a star S_i has at least three vertices, then each leaf of S_i is adjacent to an element of $\mathcal{A}_\gamma(T)$. Therefore, if any vertex $x \in \mathcal{N}_\gamma(T)$ has a neighbour $y \in \mathcal{A}_\gamma(T)$, then each γ-set D of T is of, and only of, the form $\mathcal{A}_\gamma(T) \cup \{s_1, \ldots, s_k\}$, where $s_i \in V_{S_i}$, $i = 1, \ldots, k$, which immediately implies that the number of γ-sets of T is equal to $\Pi_{i=1}^{k} |V_{S_i}|$, and hence all of them, in that specific case, can be enumerated in total time $O(|V_T| + \Pi_{i=1}^{k} |V_{S_i}|)$.

Acknowledgements
The authors would like to show their gratitude to Magda Dettlaff and Magdalena Lemańska for suggesting the topic.

REFERENCES

[1] V.E. Alekseev, R. Boliac, D.V. Korobitsyn, V.V. Lozin, *NP-hard graph problems and boundary classes of graphs*, Theor. Comput. Sci. **389** (2007), 219–236.

[2] A. Babikir, M. Dettlaff, M.A. Henning, M. Lemańska, *Independent domination subdivision in graphs*, Graphs Combin. **37** (2021), 691–709.

[3] S. Benecke, C.M. Mynhardt, *Trees with domination subdivision number one*, Australas. J. Comb. **42** (2008), 201–209.

[4] C. Berge, *Théorie des Graphes et ses Applications*, Dunod, Paris, 1958.

[5] M. Blidia, R. Lounes, *Vertices belonging to all or no minimum locating dominating set of trees*, Opuscula Math. **29** (2009), 5–14.

[6] M. Blidia, M. Chellali, S. Khelifi, *Vertices belonging to all or to no minimum double dominating sets in trees*, AKCE Int. J. Gr. Comb. **2** (2005), 1–9.

[7] H.L. Bodlaender, *Dynamic programming on graphs with bounded treewidth*, Lect. Notes Comput. Sci. **317** (1988), 105–118.

[8] E. Boros, M.C. Golumbic, V.E. Levit, *On the number of vertices belonging to all maximum stable sets of a graph*, Discrete Appl. Math. **124** (2002), 17–25.

[9] V. Bouquet, F. Delbot, Ch. Picouleau, *On the vertices belonging to all, some, none minimum dominating set*, Discrete Appl. Math. **288** (2021), 9–19.

[10] B. Brešar, F. Kardoš, J. Katrenič, G. Semanišin, *Minimum k-path vertex cover*, Discrete Appl. Math. **159** (2011), 1189–1195.

[11] T. Burton, D.P. Sumner, *γ-excellent, critically dominated, end-dominated, and dot-critical trees are equivalent*, Discrete Appl. Math. **307** (2007), 683–693.

[12] K. Cameron, P. Hell, *Independent packings in structured graphs*, Math. Program. **105** (2006), 201–213.

[13] G.J. Chang, *Algorithmic Aspects of Domination in Graphs*, [in:] *Handbook of Combinatorial Optimization*, Springer, 2013, 221–282.

[14] X.G. Chen, *Vertices contained in all minimum paired-dominating sets of a tree*, Czechoslov. Math. J. **57** (2007), 407–417.

[15] E.J. Cockayne, M.A. Henning, C.M. Mynhardt, *Vertices contained in all or in no minimum total dominating set of a tree*, Discrete Math. **260** (2003), 37–44.

[16] B. Courcelle, J.A. Makowsky, U. Rotics, *Linear time solvable optimization problems on graphs on bounded clique width*, Theory Comput. Syst. **33** (2000), 125–150.

[17] S.E. Dreyfus, A.M. Law, *The Art and Theory of Dynamic Programming*, Academic Press, New York, 1977.

[18] M. Edwards, G. MacGillivray, S. Nasserasr, *Reconfiguring minimum dominating sets: The γ-graph of a tree*, Discuss. Math. Graph Theory **38** (2018), 703–716.

[19] O. Favaron, *On a conjecture of Fink and Jacobson concerning k-domination and k-dependence*, J. Comb. Theory Ser. B **39** (1985), 101–102.

[20] G. Fricke, T. Haynes, S. Hedetniemi, S. Hedetniemi, R. Laskar, *Excellent trees*, Bull. Inst. Comb. Appl. **34** (2002), 27–38.

[21] G.H. Fricke, S.M. Hedetniemi, S.T. Hedetniemi, K.R. Hutson, *γ-Graphs of graphs*, Discuss. Math. Graph Theory **31** (2011), 517–531.

[22] P. Golovach, P. Heggernes, M.M. Kanté, D. Kratsch, Y. Villanger, *Minimal dominating sets in interval graphs and trees*, Discrete Appl. Math. **216** (2017), 162–170.

[23] G.H. Gunther, B. Hartnell, L. Markus, D. Rall, *Graphs with unique minimum dominating sets*, Congr. Numer. **101** (1994), 55–63.

[24] P.L. Hammer, P. Hansen, B. Simeone, *Veritices belonging to all or to no maximum stable sets of a graph*, SIAM J. Algebraic Discrete Methods **3** (1982), 511–522.

[25] M.A. Henning, A.J. Marcon, *Vertices contained in all or in no minimum semitotal dominating set of a tree*, Discuss. Math. Graph Theory **36** (2016), 71–93.

[26] M.A. Henning, A.J. Marcon, *Vertices contained in all or in no minimum disjunctive dominating set of a tree*, Util. Math. **105** (2017), 95–123.

[27] W.F. Klostermeyer, C.M. Mynhardt, *A dynamic domination problem in trees*, Trans. Comb. **4** (2015), 15–31.

[28] M. Krzywkowski, *Trees having many minimal dominating sets*, Inf. Process. Lett. **113** (2013), 276–279.

[29] M. Lemańska, P. Żyliński, *Reconfiguring minimum dominating sets in trees*, J. Graph Algorithms Appl. **24** (2020), 47–61.

[30] V.E. Levit, E. Mandrescu, *Combinatorial properties of the family of maximum stable sets of a graph*, Discrete Appl. Math. **117** (2002), 149–161.

[31] V.E. Levit, E. Mandrescu, *Vertices belonging to all critical sets of a graph*, SIAM J. Discrete Math. **26** (2012), 399–403.

[32] N. Meddah, M. Blidia, S. Arumugam, *Vertices contained in all or in no minimum k-dominating sets of a tree*, AKCE Int. J. Graphs Comb. **11** (2014), 105–113.

[33] C.M. Mynhardt, *Vertices contained in every minimum dominating set of a tree*, J. Graph Theory **31** (1999), 155–265.

[34] C.M. Mynhardt H.C. Swart, E. Ungerer, *Excellent trees and secure domination*, Util. Math. **67** (2005), 255–267.

[35] O. Ore, *Theory of Graphs*, American Mathematical Society Colloquium Publications vol. XXXVIII, AMS, Providence, 1962.

[36] Y. Orlovich, A. Dolgui, G. Finke, V. Gordon, F. Werner, *The complexity of dissociation set problems in graphs*, Discrete Appl. Math. **159** (2011), 1352–1366.

[37] J. Petr, J. Portier, L. Versteegen, *On the number of minimum dominating sets and total dominating sets in forests*, J. Graph Theory **106** (2024), 976–993.

[38] G. Rote, *The maximum number of minimal dominating sets in a tree*, SODA (2019), 1201–1214.

[39] V. Samodivkin, *The bondage number of graphs: good and bad vertices*, Discuss. Math. Graph Theory **28** (2008), 453–462.

[40] J. Tu, L. Zhang, J. Du, R. Lang, *Polynomial time recognition of vertices contained in all (or no) maximum dissociation sets of a tree*, AIMS math. **7** (2022), 569–578.

[41] M. Yannakakis, *Node-deletion problems on bipartite graphs*, SIAM J. Comput. **10** (1981), 310–327.

[42] https://codeforces.com/contest/1092/problem/F (accessed: 13.10.2025)

[43] https://www.geeksforgeeks.org/competitive-programming/dp-on-trees-for-competitive-programming/(accessed 13.10.2025)

[44] https://usaco.guide/gold/all-roots (accessed: 13.10.2025)

Radosław Ziemann (corresponding author)
radoslaw.ziemann@ug.edu.pl
https://orcid.org/0000-0001-9659-0911

University of Gdańsk
Department of Combinatorial Optimization
80-308 Gdańsk, Poland

Paweł Żyliński
pawel.zylinski@ug.edu.pl
https://orcid.org/0000-0001-6378-7742

University of Gdańsk
Department of Combinatorial Optimization
80-308 Gdańsk, Poland

Received: May 15, 2025.
Revised: October 10, 2025.
Accepted: November 5, 2025.