Weighted efficient domination problem on some perfect graphs

Chin Lung Lu\textsuperscript{a}, Chuan Yi Tang\textsuperscript{b,}\textsuperscript{*}

\textsuperscript{a}National Center for High-Performance Computing, P.O. Box 19-136, Hsinchu, Taiwan, ROC
\textsuperscript{b}Department of Computer Science, National Tsing Hua University, Hsinchu, 30043 Taiwan, ROC

Received 16 July 1998; received in revised form 29 June 1999; accepted 30 October 2000

Abstract

Given a simple graph \( G = (V, E) \), a vertex \( v \in V \) is said to dominate itself and all vertices adjacent to it. A subset \( D \) of \( V \) is called an efficient dominating set of \( G \) if every vertex in \( V \) is dominated by exactly one vertex in \( D \). The efficient domination problem is to find an efficient dominating set of \( G \) with minimum cardinality. Suppose that each vertex \( v \in V \) is associated with a weight. Then, the weighted efficient domination problem is to find an efficient dominating set with the minimum weight in \( G \). In this paper, we show that the efficient domination problem is NP-complete for planar bipartite graphs and chordal bipartite graphs. Assume that a permutation diagram of a bipartite permutation graph and a one-vertex-extension ordering of a distance-hereditary graph are given in advance. Then, we give \( O(|V|) \) time algorithms for the weighted efficient domination problem on bipartite permutation graphs and distance-hereditary graphs. © 2002 Elsevier Science B.V. All rights reserved.

Keywords: Algorithms; Efficient domination; Planar bipartite graphs; Chordal bipartite graphs; Bipartite permutation graphs; Distance-hereditary graphs

1. Introduction

Let \( G = (V, E) \) be a simple graph, i.e., finite, undirected, and loopless graph without multiple edges. The open neighborhood \( N(v) \) of the vertex \( v \) consists of the set of vertices adjacent to \( v \), i.e., \( N(v) = \{ u \in V \mid (u, v) \in E \} \), and the closed neighborhood of \( v \) is \( N[v] = \{ v \} \cup N(v) \). For any two vertices \( u, v \in V \), the distance \( d(u, v) \) of vertices \( u \) and \( v \) is the minimum length of a path between \( u \) and \( v \). Define \( d(u, v) = \infty \) if there exists no path between vertices \( u \) and \( v \). A vertex \( v \in V \) is said to dominate all vertices in \( N[v] \). A subset \( D \) of \( V \) is called a dominating set if every vertex of \( V \) is dominated by at least one vertex in \( D \). A dominating set \( D \) of \( G \) is efficient if every vertex in \( V \) is dominated by exactly one vertex of \( D \), or equivalently, if for any two vertices

\textsuperscript{*} Corresponding author.
E-mail address: cytang@cs.nthu.edu.tw (Chuan Yi Tang).
u, v ∈ D, d(u, v) ≥ 3. We say that an efficient dominating set D of G efficiently dominates every vertex in V. Note that not all graphs have efficient dominating sets. Those graphs that have an efficient dominating set include path P_n for all n, cycle C_n if and only if n ≡ 0 (mod 3), complete bipartite graph K_{m,n} if and only if m = 1 or n = 1, and complete graph K_n for all n [3]. Whether an efficient dominating set exists for meshes, tori, trees, dags, series-parallel graphs, hypercubes, cube-connected cycles, cube-connected paths, and de Bruijn graphs is considered in [27,32].

In this paper, we study the efficient domination problem which is to find an efficient dominating set of G with minimum cardinality if such a set exists. It is not difficult to see that \( D = \{v_1, v_2, \ldots, v_k\} \) is an efficient dominating set of G if and only if \( \{N[v_1], N[v_2], \ldots, N[v_k]\} \) is a partition of V [22]. In [3], Bange, Barkauskas and Slater showed that if G has an efficient dominating set, then the cardinality of any efficient dominating set equals the domination number \( \gamma(G) \) of G, where domination number \( \gamma(G) \) is the cardinality of a minimum dominating set of G. In other words, all efficient dominating sets of G have the same cardinality and hence the efficient domination problem is equivalent to find an efficient dominating set in G. Suppose that each vertex \( v \in V \) is associated with a real number \( w(v) \), called the weight of v. The weighted efficient domination problem is to find an efficient dominating set \( D \) of G such that the weight \( w(D) \) of \( D \) is minimum, where \( w(D) = \sum_{v \in D} w(v) \).

Efficient domination was introduced by Bange et al. [2,23] when they constructively characterized trees with disjoint dominating sets of several types. There are many interesting applications for efficient domination in coding theory [4,5,21], graph embedding [30,31], facility location on geographical area [33–36], and resource allocation in parallel processing system [26,27,32], often with different terminologies. In fact, an earlier work using the same concept of efficient domination was proposed by Biggs who studied the perfect code problem. He introduced perfect d-codes and his perfect 1-code is identical to efficient domination [4,5,21]. Then, Weichsel who investigated the graph embedding problem introduced perfect domination [30,31] and his independent perfect domination is equivalent to efficient domination [33–36]. Later, Livingston and Stout proposed perfect d-domination, which is equal to perfect d-codes, when they studied resource allocation and placement in parallel computers [27,32]. In [17], Fellows and Hoover called efficient domination as perfect domination and studied its algorithmic complexity on some subclasses of planar graphs.

There is an extensive number of papers concerning the algorithmic complexity of the weighted efficient domination problem in graphs. Bange, Barkauskas and Slater proved that the efficient domination problem is NP-complete on general graphs and gave an \( O(|V|^3) \) time algorithm for this problem on trees [3]. Fellows and Hoover showed that the efficient domination problem is NP-complete on planar graphs of maximum degree three [17]. Yen and Lee proved that the independent perfect domination problem is NP-complete on bipartite graphs and chordal graphs [33,36]. They also gave \( O(|V|^2 + |E|) \) time algorithms for the weighted case on series-parallel graphs and block graphs [33,36]. Chang and Liu proposed \( O(|V| + |E|) \) time algorithms for solving the weighted independent perfect domination problem on split graphs [11] and interval
Fig. 1. Hierarchy of some graph classes and their previously known complexity results.

graphs [12]. They also generalized their interval algorithm to an $O(|V||E| + |V|^2)$ time algorithm for circular-arc graphs. Chang et al. [13] presented an $O(|V||E|)$ time algorithm for the weighted independent perfect domination problem on cocomparability graphs. With some modifications, their algorithm yields an $O(|V| + |E|)$ time algorithm on interval graphs. However, their result on cocomparability graphs was later improved to $O(|V|^2)$ by Chang [9]. In [25], Liang, Lu and Tang gave an $O(|V| + |E|)$ time algorithm for the weighted efficient domination problem on permutation graphs and generalized it to an $O(|V|\log \log |V| + |E|)$ time algorithm on trapezoid graphs, where $|E|$ denotes the number of edges in the complement of $G$. In other words, the efficient domination problem is NP-complete on planar graphs of maximum degree three [17], bipartite graphs [33,36] and chordal graphs [33,36], and its weighted case can be solved in polynomial or even in linear time on trees [3], series–parallel graphs [33], block graphs [33,36], split graphs [11], interval graphs [12,13], cocomparability graphs [9,13], circular-arc graphs [12], permutation graphs [25] and trapezoid graphs [25]. Fig. 1 shows the hierarchy of some special classes of graphs and their previously known complexity results on the weighted efficient domination problem, where “?” represents the complexity being unknown. Definitions of graph classes not found in this paper are standard and may be found in [7,19].
In this paper, we show that the efficient domination problem is NP-complete when restricted to planar bipartite graphs and chordal bipartite graphs. In addition, for the weighted efficient domination problem, we give an $O(|V|)$ time algorithm for a bipartite permutation graph given with a permutation diagram and an $O(|V|)$ time algorithm for a distance-hereditary graph given with a one-vertex-extension ordering.

2. NP-completeness results

A graph is planar if it can be drawn on the plane such that no two edges cross each other. A graph is bipartite if its vertex set can be partitioned into two subsets such that no edge joins two vertices in the same set (i.e., two independent sets). Planar bipartite graphs are exactly those graphs that are both planar and bipartite. Chordal bipartite graphs are bipartite graphs in which every cycle of length greater than four has a chord, i.e., an edge between two non-consecutive vertices of the cycle. For more detailed information on the properties and the applications of chordal bipartite graphs, the reader is referred to [7,19]. In this section, we will first show that Problem ED on planar bipartite graphs is NP-complete and then show that Problem ED on chordal bipartite graphs is also NP-complete.

**Problem ED (Efficient domination).**

*Instance*: A graph $G = (V,E)$.

*Question*: Does $G$ have an efficient dominating set?

**X3C (Exact cover by 3-sets).**

*Instance*: A finite set $X$ with $|X| = 3n$ and a collection $\mathcal{S}$ of 3-element subsets of $X$ with $|\mathcal{S}| = m$.

*Question*: Does $\mathcal{S}$ contain an exact cover for $X$, i.e., a subcollection $\mathcal{S}' \subseteq \mathcal{S}$ such that every element of $X$ occurs in exactly one member of $\mathcal{S}'$?

It is well known that X3C is NP-complete [18]. Note that each instance of X3C, say $X = \{x_1, x_2, \ldots, x_{3n}\}$ and $\mathcal{S} = \{S_1, S_2, \ldots, S_m\}$, can be associated with a bipartite graph $G_i = (V_i, E_i)$, where $V_i = X \cup \mathcal{S}$ and $E_i = \{(x_i, S_j) | 1 \leq i \leq 3n, 1 \leq j \leq m$ and $x_i \in S_j\}$. If the associated bipartite graph $G_i$ is planar, then the problem is said to be planar exact cover by 3-sets (planar X3C). In [16], Dyer and Frieze showed that planar X3C is NP-complete. In the following, we show that Problem ED on planar bipartite graphs is NP-complete by reducing from planar X3C. This proof is based on a construction proposed by Yen and Lee [36] which showed that Problem ED on bipartite graphs is NP-complete.

**Theorem 2.1.** Problem ED on planar bipartite graphs is NP-complete.

**Proof.** Obviously, the problem is in NP. In the following, we show that planar X3C is polynomially reducible to this problem. Let $X = \{x_1, x_2, \ldots, x_{3n}\}$ and $\mathcal{S} = \{S_1, S_2, \ldots, S_m\}$
Fig. 2. $G_S$ for $S = \{S_1, S_2, S_3\} = \{\{x_1, x_2, x_4\}, \{x_2, x_4, x_6\}, \{x_3, x_5, x_6\}\}$.

be an instance of planar X3C. Then, we construct a planar bipartite graph $G_S = (V_S, E_S)$ as follows:

$V_S = \{x_1, x_2, \ldots, x_{3n}\} \cup \{S_1, S_2, \ldots, S_m\} \cup \{a_1, a_2, \ldots, a_m\}$,

$E_S = \{(x_i, S_j) | 1 \leq i \leq 3n, 1 \leq j \leq m \text{ and } x_i \in S_j\} \cup \{(S_j, a_j) | 1 \leq j \leq m\}$.

See Fig. 2 for an example with $X = \{x_1, x_2, \ldots, x_6\}$ and $S = \{S_1, S_2, S_3\} = \{\{x_1, x_2, x_4\}, \{x_2, x_4, x_6\}, \{x_3, x_5, x_6\}\}$. Note that the subgraph of $G_S$ induced by $X \cup S$ is bipartite and planar. Hence, $G_S$ is a planar bipartite graph and its construction takes polynomial time.

Next, we claim that $S$ has an exact cover $S'$ if and only if $G_S$ has an efficient dominating set $D$. First, suppose that $S$ has an exact cover $S'$. Then, we define $D = \{S_j | S_j \in S'\} \cup \{a_j | S_j \not\in S'\}$. It is easy to verify that $D$ is an efficient dominating set of $G_S$.

Conversely, suppose that $G_S$ has an efficient dominating set $D$. Note that $D \cap \{S_j, a_j\} \neq \emptyset$ for each $1 \leq j \leq m$; otherwise, $D$ does not dominate $a_j$. Let $S'$ be defined by $S_j \in S'$ if $S_j \in D$. It is clear that $S'$ is an exact cover of $S$.

In the following, we show that Problem ED on chordal bipartite graphs is NP-complete by reducing from one-in-three 3SAT. Note that one-in-three 3SAT is well known to be NP-complete [18].

**One-in-three 3SAT**

*Instance:* Set $U$ of boolean $n$ variables, collection $C$ of $m$ clauses over $U$ such that each clause has exactly three literals.

*Question:* Is there a truth assignment $t : U \rightarrow \{true, false\}$ for $C$ such that each clause in $C$ has exactly one true literal?

Given an instance of one-in-three 3SAT, say $U = \{u_1, u_2, \ldots, u_n\}$ and a formula $C = C_1 \land C_2 \land \cdots \land C_m$ with each clause $C_j$ containing three literals $l_{j1}, l_{j2}$ and $l_{j3}$,
we construct a chordal bipartite graph \( G_\emptyset = (V_\emptyset, E_\emptyset) \) using the following three steps:

(We assume that no clause contains both a literal and its negation because this clause is always true and can be omitted.)

1. For each variable \( u_i \), where \( 1 \leq i \leq n \), we construct the subgraph \( G(u_i) \) of \( G_\emptyset \) as shown in Fig. 3(a).

2. For each clause \( C_j \), where \( 1 \leq j \leq m \), we construct the subgraph \( G(C_j) \) of \( G_\emptyset \) as shown in Fig. 3(b) with \( v_{jk} = a_{ij} \) iff \( l_{jk} = u_i \), and \( v_{jk} = \bar{a}_{ij} \) iff \( l_{jk} = \bar{u}_i \) for all \( 1 \leq k \leq 3 \).

3. Finally, we add all possible edges between \( \mathcal{E} \cup \mathcal{X} \) and \( \mathcal{F} \cup \mathcal{Y} \) such that they form a complete bipartite subgraph of \( G_\emptyset \), where \( \mathcal{E} = \{ e_{ij} | 1 \leq i \leq n \text{ and } 1 \leq j \leq m \} \), \( \mathcal{X} = \{ x_j | 1 \leq j \leq m \} \), \( \mathcal{F} = \{ f_{ij} | 1 \leq i \leq n \text{ and } 1 \leq j \leq m \} \) and \( \mathcal{Y} = \{ y_{ij} | 1 \leq i \leq n \text{ and } 2 \leq j \leq m \} \).

Before proceeding our discussion, we define the following notation:

- \( \mathcal{A} = \{ a_{ij}, \bar{a}_{ij} | 1 \leq i \leq n \text{ and } 1 \leq j \leq m \} \) and \( \mathcal{B} = \{ b_{ij}, \bar{b}_{ij} | 1 \leq i \leq n \text{ and } 1 \leq j \leq m \} \).
- \( \mathcal{S} = \{ s_{ij} | 1 \leq i \leq n \text{ and } 2 \leq j \leq m \} \).
- For each \( 1 \leq i \leq n \), \( \mathcal{A}_i = \{ a_{ij} | 1 \leq j \leq m \} \) and \( \mathcal{B}_i = \{ b_{ij} | 1 \leq j \leq m \} \).
- For each \( 1 \leq i \leq n \), \( \mathcal{S}_i = \{ s_{ij} | 1 \leq j \leq m \} \).

**Claim 2.1.** \( G_\emptyset \) is a chordal bipartite graph.

**Proof.** Obviously, \( G_\emptyset \) is a bipartite graph. Suppose that there is a cycle \( \Phi \) of length six or more in \( G_\emptyset \). Then, we distinguish the following four cases.

**Case 1:** There is an edge \( (a, b) \) in \( \Phi \), where \( a \in \mathcal{A} \) and \( b \in \mathcal{B} \). Let \( v' \neq b \) and \( v'' \neq a \) be the neighbors of \( a \) and \( b \) in \( \Phi \), respectively. Then, by the construction of \( G_\emptyset \), we have \( v' \in \mathcal{E} \cup \mathcal{X} \) and \( v'' \in \mathcal{F} \cup \mathcal{Y} \). Since \( \mathcal{E} \cup \mathcal{X} \) and \( \mathcal{F} \cup \mathcal{Y} \) form a complete bipartite subgraph, \( (v', v'') \in E_\emptyset \), i.e., there is a chord in \( \Phi \).

**Case 2:** There are two non-consecutive vertices \( a \) and \( b \) in \( \Phi \), where \( a \in \mathcal{A} \) and \( b \in \mathcal{B} \). Then, by the construction of \( G_\emptyset \), the neighbors of \( a \) and \( b \) in \( \Phi \) are elements of \( \mathcal{E} \cup \mathcal{X} \) and \( \mathcal{F} \cup \mathcal{Y} \), respectively. Since \( \mathcal{E} \cup \mathcal{X} \) and \( \mathcal{F} \cup \mathcal{Y} \) form a complete bipartite subgraph, there is a chord in \( \Phi \).

**Case 3:** There is no vertex of \( \mathcal{A} \) in \( \Phi \). According to the construction of \( G_\emptyset \), cycle \( \Phi \) alternates between \( \mathcal{B} \cup \mathcal{E} \cup \mathcal{X} \) and \( \mathcal{F} \cup \mathcal{Y} \) only. Note that any path that alternates
between \( B \) and \( F \cup \mathcal{Y} \) cannot form a cycle. In other words, there is at least a vertex \( v \in E \cup X \) in \( \Phi \). Since \( v \) is adjacent to all vertices in \( F \cup \mathcal{Y} \), there is a chord in \( \Phi \).

**Case 4:** There is no vertex of \( B \) in \( \Phi \). According to the construction of \( G_{\mathcal{G}} \), cycle \( \Phi \) alternates between \( E \cup X \) and \( A \cup F \cup \mathcal{Y} \) only. Note that any path that alternates between \( A \) and \( E \cup X \) cannot form a cycle. In other words, there is at least a vertex \( v \in F \cup \mathcal{Y} \) in \( \Phi \). Since \( v \) is adjacent to all vertices in \( E \cup X \), there is a chord in \( \Phi \).

**Claim 2.2.** Let \( D \) be an efficient dominating set in \( G_{\mathcal{G}} \). Then, \( D \cap \mathcal{E} = \emptyset \) and \( D \cap \mathcal{F} = \emptyset \).

**Proof.** Suppose that \( D \cap \mathcal{E} \neq \emptyset \) and let \( e_{ij} \in \mathcal{E} \) be in \( D \). Then, no neighbor \( v \) of \( b_{ij} \) is in \( D \) since \( d(v, e_{ij}) = 1 \). Hence, \( D \) contains \( b_{ij} \) to efficiently dominate it. However, \( d(b_{ij}, e_{ij}) = 2 \), a contradiction. In other words, \( D \cap \mathcal{E} = \emptyset \). Similarly, we have \( D \cap \mathcal{F} = \emptyset \).

**Claim 2.3.** Let \( D \) be an efficient dominating set in \( G_{\mathcal{G}} \). Then, \( \mathcal{R} \subseteq D \) and \( \mathcal{S} \subseteq D \).

**Proof.** Suppose that \( \mathcal{R} \not\subseteq D \) and let \( r_j \in \mathcal{R} \) be not in \( D \). Then, \( D \) contains \( g_j \) and \( z_j \) to efficiently dominate them. However, \( d(g_j, z_j) = 2 \), a contradiction. In other words, \( \mathcal{R} \subseteq D \). Similarly, we have \( \mathcal{S} \subseteq D \).

**Claim 2.4.** Let \( D \) be an efficient dominating set in \( G_{\mathcal{G}} \). Then, \( D \cap \mathcal{X} = \emptyset \) and \( D \cap \mathcal{Y} = \emptyset \).

**Proof.** Suppose that \( D \cap \mathcal{X} \neq \emptyset \) and let \( x_j \in \mathcal{X} \) be in \( D \). By Claim 2.3, \( r_j \) is also in \( D \). However, \( d(r_j, x_j) = 2 \), a contradiction. In other words, \( D \cap \mathcal{X} = \emptyset \). Similarly, we have \( D \cap \mathcal{Y} = \emptyset \).

**Claim 2.5.** Let \( D \) be an efficient dominating set in \( G_{\mathcal{G}} \). Then, \( a_{ij} \in D \) if and only if \( b_{ij} \in D \), where \( 1 \leq i \leq n \) and \( 1 \leq j \leq m \).

**Proof.** First, suppose that \( a_{ij} \in D \) and \( b_{ij} \not\in D \). By Claim 2.2, \( e_{ij} \not\in D \). If there is a vertex \( x_{j'} \) adjacent to \( \tilde{a}_{ij} \), then \( x_{j'} \) is not in \( D \) according to Claim 2.4. In other words, no neighbor of \( \tilde{a}_{ij} \) is in \( D \). Hence, \( D \) contains \( \tilde{a}_{ij} \) to efficiently dominate it. However, \( d(\tilde{a}_{ij}, a_{ij}) = 2 \), a contradiction.

Conversely, suppose that \( b_{ij} \in D \) and \( a_{ij} \not\in D \). By Claim 2.2, \( f_{ij} \not\in D \). If there exists \( y_{ij} \) such that \( (y_{ij}, \tilde{b}_{ij}) \in E_{\mathcal{G}} \), then \( y_{ij} \) is not in \( D \) according to Claim 2.4. In other words, no neighbor of \( \tilde{b}_{ij} \) is in \( D \). Hence, \( D \) contains \( \tilde{b}_{ij} \) to efficiently dominate it. However, \( d(\tilde{b}_{ij}, b_{ij}) = 2 \), a contradiction.

Similar to Claim 2.5, we have the following lemma.

**Claim 2.6.** Let \( D \) be an efficient dominating set in \( G_{\mathcal{G}} \). Then, \( \tilde{a}_{ij} \in D \) if and only if \( \tilde{b}_{ij} \in D \), where \( 1 \leq i \leq n \) and \( 1 \leq j \leq m \).
Claim 2.7. Let D be an efficient dominating set in \( G_\ell \). For each \( 1 \leq i \leq n \), either all vertices of \( \mathcal{A}_i \cup \mathcal{B}_i \) or all vertices of \( \tilde{\mathcal{A}}_i \cup \tilde{\mathcal{B}}_i \) are in D.

Proof. By Claims 2.2 and 2.4, no vertex of \( \mathcal{D} \cup \mathcal{F} \cup \mathcal{X} \cup \mathcal{Y} \) is in D. For each \( 1 \leq j \leq m \), D contains exactly one of \( a_{ij} \) and \( \tilde{b}_{ij} \) to efficiently dominate \( b_{ij} \). By Claims 2.5 and 2.6, either \( \{a_{ij}, b_{ij}\} \subseteq D \) or \( \{\tilde{a}_{ij}, \tilde{b}_{ij}\} \subseteq D \). Suppose that there are two consecutive numbers \( j' \) and \( j'' \) such that either (1) \( a_{ij'}; b_{ij'}, \tilde{a}_{ij''} \) and \( \tilde{b}_{ij''} \) are in D or (2) \( \tilde{a}_{ij'}, \tilde{b}_{ij''}; a_{ij''} \) and \( b_{ij''} \) are in D. In the first case, \( d(b_{ij'}, \tilde{b}_{ij''}) = 2 \) and a contradiction arises. In the second case, \( s_{ij''} \) is in D by Claim 2.3 and \( q_{ij''} \notin D \) since \( d(q_{ij''}, s_{ij''}) = 1 \). Note that no vertex of \( \mathcal{D} \cup \mathcal{F} \cup \mathcal{X} \cup \mathcal{Y} \) is in D. In other words, no vertex in \( N[y_{ij''}] \) is in D. As a result, \( y_{ij''} \) not in D cannot be dominated by any vertex in D, a contradiction. Therefore, either all vertices of \( \mathcal{A}_i \cup \mathcal{B}_i \) or all vertices of \( \tilde{\mathcal{A}}_i \cup \tilde{\mathcal{B}}_i \) are in D.

Theorem 2.2. Problem ED on chordal bipartite graphs is NP-complete.

Proof. Obviously, the problem is in NP. In the following, we show that one-in-three 3SAT is polynomially reducible to this problem. Given an instance of one-in-three 3SAT, say \( \bar{U} = \{u_1, u_2, \ldots, u_n\} \) and a formula \( \mathcal{C} = C_1 \land C_2 \land \cdots \land C_m \) with each clause \( C_j \) containing three literals \( l_{j1}, l_{j2}, l_{j3} \), we construct a chordal bipartite graph \( G_\ell = (V_\ell, E_\ell) \) as mentioned previously. The construction of \( G_\ell \) takes polynomial time.

We next show that \( \mathcal{C} \) has a satisfying truth assignment if and only if \( G_\ell \) has an efficient dominating set. First, suppose that \( \mathcal{C} \) has a satisfying truth assignment such that exactly one of \( l_{j1}, l_{j2}, l_{j3} \) is true for each \( 1 \leq j \leq m \). Define \( D \subseteq V_\ell \) as follows. Let \( \mathcal{A} \cup \mathcal{S} \subseteq D \). For each variable \( u_i \), if \( t(u_i) = true \), then all vertices of \( \mathcal{A}_i \cup \mathcal{B}_i \) are included in D; otherwise, all vertices of \( \tilde{\mathcal{A}}_i \cup \tilde{\mathcal{B}}_i \) are included in D. It is easy to verify that \( D \) is an efficient dominating set in \( G_\ell \).

Conversely, suppose that \( G_\ell \) has an efficient dominating set \( D \). Note that for each \( 1 \leq i \leq n \), either all vertices of \( \mathcal{A}_i \cup \mathcal{B}_i \) or all vertices of \( \tilde{\mathcal{A}}_i \cup \tilde{\mathcal{B}}_i \) are in \( D \) according to Claim 2.7. Let \( t: U \rightarrow \{true, false\} \) be defined by \( t(u_i) = true \) if and only if \( \mathcal{A}_i \cup \mathcal{B}_i \subset D \). Consider vertex \( x_j \), where \( 1 \leq j \leq m \). We have \( x_j \notin D \) and \( r_j \in D \) according to Claims 2.4 and 2.3, respectively. \( p_j \notin D \) since \( d(p_j, r_j) = 1 \) and no vertex of \( \mathcal{S} \cup \mathcal{Y} \) adjacent to \( x_j \) is in \( D \) by Claims 2.2 and 2.4. Therefore, \( D \) contains exactly one of \( v_{ij}, v_{j2}, v_{j3} \) to efficiently dominate \( x_j \), which implies that each clause \( C_j \) has exactly one true literal. In other words, \( t \) is a one-in-three satisfying truth assignment.

3. An \( O(|V|) \) algorithm on bipartite permutation graphs

Note that the weighted domination problem on permutation graphs can be solved in \( O(|V| + |E|) \) time if a permutation diagram is given [25]. Since bipartite permutation graphs form a subclass of permutation graphs, the same problem on bipartite permutation graphs is solvable in \( O(|V| + |E|) \) time. In this section, we will improve this result to \( O(|V|) \) if a permutation diagram is given.
A graph \( G = (V, E) \) is a permutation graph if there are a labelling \( L = \{1, 2, \ldots, |V|\} \) of the vertices in \( V \) and a permutation \( \pi = [\pi(1), \pi(2), \ldots, \pi(|V|)] \) of \( L \) such that \((i, j) \in E \) if and only if \((i - j)(\pi^{-1}(i) - \pi^{-1}(j)) < 0\), where \( \pi^{-1}(i) \) denotes the position of number \( i \) in \( \pi \). A permutation graph is often represented by its corresponding permutation diagram (i.e., intersection model). The permutation diagram consists of two horizontal parallel channels, named top channel and bottom channel, respectively. We put the numbers 1, 2, \ldots, \(|V|\) on the top channel in order from left to right, and put the numbers \( \pi(1), \pi(2), \ldots, \pi(|V|) \) on the bottom channel in the same way. Then for each \( i \), we draw a straight line joining the two \( i \)'s, where one on the top channel and the other on the bottom channel, and label each such line by the same number \( i \). For example, Fig. 4 shows a permutation graph and its permutation diagram. Note that lines \( i \) and \( j \) intersect in the permutation diagram if and only if vertices \( i \) and \( j \) of the corresponding permutation graph are adjacent.

We use \( G = (A, B, E) \) to denote a bipartite graph with two independent vertex sets \( A \) and \( B \) such that \( A \cup B = V \) and \( A \cap B = \emptyset \). A bipartite permutation graph is a permutation graph which is bipartite [8,28,29]. A strong ordering of the vertices of \( G = (A, B, E) \) consists of an ordering of \( A \) and an ordering of \( B \) such that for all \((a, b), (a', b') \in E\), where \( a, a' \in A \) and \( b, b' \in B \), \( a < a' \) and \( b' < b \) imply \((a, b'), (a', b) \in E\). An ordering of the vertices of \( A \) has the adjacency property if for each vertex \( b \in B \), \( N(b) \) consists of vertices which are consecutive in the ordering of \( A \). An ordering of the vertices of \( A \) has the enclosure property if for every two vertices \( b, b' \in B \) with \( N(b) \subset N(b') \), vertices in \( N(b') \setminus N(b) \) occur consecutively in the ordering of \( A \).

**Lemma 3.1** (Spinrad [28]). Let \( G = (A, B, E) \) be a bipartite graph. Then, the following statements are equivalent:

1. \( G \) is a bipartite permutation graph.
2. There is a strong ordering of \( A \cup B \).
3. There exists an ordering of \( A \) which has the adjacency and enclosure properties.

In [28], Spinrad, Brandstädt and Stewart gave a linear time algorithm for recognizing whether a given graph is a bipartite permutation graph and producing a permutation diagram.
diagram if so. They also claimed that the orderings of $A$ and $B$ in which vertices are ordered by their position in the top channel of a permutation diagram constitute a strong ordering [28]. According to this claim, a strong ordering of vertices can be produced in $O(|V|)$ time from a permutation diagram. Note that given a strong ordering of $A \cup B$, both $A$ and $B$ have the adjacency and enclosure properties if all isolated vertices of $G$ appear at the beginning of the orderings of $A$ and $B$ [8].

For simplicity of illustrating algorithms, we assume that the given bipartite permutation graph $G = (A, B, E)$ is connected and a permutation diagram of $G$ is also given. Let $A = \{a_1, a_2, \ldots, a_m\}$ and $B = \{b_1, b_2, \ldots, b_n\}$ be the vertices of $A$ and $B$ in the strong ordering such that $a_i < a_{i'}$ if and only if $1 \leq i < i' \leq m$ and $b_j < b_{j'}$ if and only if $1 \leq j < j' \leq n$, respectively. For each vertex $v \in A \cup B$, $a_i \in A$ and $b_j \in B$, we define the following notation:

- $s(v) = \min N(v)$, i.e., the smallest vertex adjacent to $v$.
- $l(v) = \max N(v)$, i.e., the largest vertex adjacent to $v$.
- $V(a_i) = \{a_k \in A \mid a_k \leq a_i\} \cup \{b_k \in B \mid b_k \leq l(a_i)\}$.
- $V(b_j) = \{b_k \in B \mid b_k \leq b_j\} \cup \{a_k \in A \mid a_k \leq l(b_j)\}$.
- $G(a_i) = \text{the subgraph of } G \text{ induced by } V(a_i)$.
- $G(b_j) = \text{the subgraph of } G \text{ induced by } V(b_j)$.
- $ED(a_i) = \text{a minimum weighted efficient dominating set of } G(a_i)$ with the condition that $a_i$ is in $D$.
- $ED(b_j) = \text{a minimum weighted efficient dominating set of } G(b_j)$ with the condition that $b_j$ is in $D$.

**Lemma 3.2.** If $G$ has an efficient dominating set $D$, then $D \cap \{a_m, b_n\} \neq \emptyset$.

**Proof.** Suppose that $D$ is an efficient dominating set of $G$ and $D \cap \{a_m, b_n\} = \emptyset$. Then, there are two vertices $a_i$ with $1 \leq i < m$ and $b_j$ with $1 \leq j < n$ in $D$ such that $(a_i, b_j) \in E$ and $(a_m, b_j) \in E$. By the strong ordering of vertices, we have $(a_i, b_j) \in E$, a contradiction. ☐

**Lemma 3.3.** $(a_1, b_1) \in E$ and $(a_m, b_n) \in E$.

**Proof.** Suppose that $(a_1, b_1) \not\in E$. Since $G$ is connected, there are vertices $a_i \in A$ with $i > 1$ and $b_j \in B$ with $j > 1$ such that $(a_1, b_j)$ and $(a_i, b_1)$ are in $E$. By the strong ordering of vertices, we have $(a_1, b_j) \in E$, a contradiction. Similarly, $(a_m, b_n) \in E$. ☐

According to Lemmas 3.2 and 3.3, if $G$ has an efficient dominating set $D$, then $|D \cap \{a_m, b_n\}| = 1$, i.e., $D$ contains either $a_m$ or $b_n$. Hence, $\min\{ED(a_m), ED(b_n)\}$ is a minimum weighted efficient dominating (MWED) set of $G$. The following four properties are clear and useful for the design of our algorithms:

- **(P1)** If $a_i < a_{i'}$, then $s(a_i) \leq s(a_{i'})$ and $l(a_i) \leq l(a_{i'})$.
- **(P2)** If $b_j < b_{j'}$, then $s(b_j) \leq s(b_{j'})$ and $l(b_j) \leq l(b_{j'})$.
- **(P3)** $(a_i, b_j)$ is an edge in $G$ for $s(a_i) \leq b_j \leq l(a_i)$. 

---

Lemma 3.4. \( ED(a_1) = \{ a_1 \} \) and \( ED(b_1) = \{ b_1 \} \).

We use \textit{null} to denote a set which does not exist, and let \( S \cup \textit{null} = \textit{null} \) for any \( S \subseteq V \).

Lemma 3.5. Let \( i \geq 2 \) and \( s(a_i) = b_j \). Then,

\[
ED(a_i) = \begin{cases} 
\text{null} & \text{if } j = 1 \text{ or } (a_{i-1}, b_{j-1}) \notin E, \\
ED(b_{j-1}) \cup \{ a_i \} & \text{if } j \neq 1 \text{ and } (a_{i-1}, b_{j-1}) \in E.
\end{cases}
\]

Proof. We first claim that \( (a_{i-1}, b_j) \in E \). Suppose that \( (a_{i-1}, b_j) \notin E \). Then, we have the following two cases.

Case 1: \( s(a_{i-1}) > b_j \). Note that \( (a_{i-1}, s(a_{i-1})) \in E \) and \( (a_{i-1}, b_j) \in E \). By the strong ordering of vertices, we have \( (a_{i-1}, b_j) \in E \), a contradiction.

Case 2: \( l(a_{i-1}) < b_j \). According to (P1), we have \( l(a_{i'}) < b_j \) for all \( a_{i'} \) with \( i' \leq i - 1 \). If \( j = 1 \), then \( G \) is disconnected and a contradiction arises. Suppose that \( j \neq 1 \). Then, \( l(b_{j-1}) < a_i \) since \( s(a_i) = b_j \). According to (P2), we have \( l(b_{j'}) < a_i \) for all \( b_{j'} \) with \( j' \leq j - 1 \). As a result, \( G \) is disconnected, a contradiction.

Hence, \( (a_{i-1}, b_j) \in E \) and \( l(a_{i-1}) \leq l(a_i) \) by (P1). According to (P3), any \( b_k \) with \( b_j \leq b_k \leq l(a_{i-1}) \) is adjacent to \( a_i \) and hence \( b_k \notin ED(a_i) \) and \( a_{i-1} \notin ED(a_i) \). To efficiently dominate \( a_{i-1} \), \( ED(a_i) \) must contain a vertex \( b_p \) in \( B' \), where \( B' = N(a_{i-1}) \setminus \{ b_k | b_j \leq b_k \leq l(a_{i-1}) \} \). Consider the case in which \( j = 1 \) or \( (a_{i-1}, b_{j-1}) \notin E \). If \( j = 1 \), then \( B' = \emptyset \) and hence \( ED(a_i) = \textit{null} \). If \( (a_{i-1}, b_{j-1}) \notin E \), then \( s(a_{i-1}) = b_j \) and hence \( B' = \emptyset \) and \( ED(a_i) = \textit{null} \). Consider the case in which \( j \neq 1 \) and \( (a_{i-1}, b_{j-1}) \in E \). Then, \( b_{j-1} \in B' \) and \( l(b_{j-1}) = a_{i-1} \) since \( s(a_i) = b_j \). Suppose that \( b_p \neq b_{j-1} \). Then, \( ED(a_i) \) contains a vertex \( a_q \) with \( s(b_{j-1}) \leq a_q < a_{i-1} \) to efficiently dominate \( b_{j-1} \). By the strong ordering of vertices, we have \( (a_q, b_p) \in E \), a contradiction. In other words, \( b_{j-1} \in ED(a_i) \) and hence \( ED(a_i) = ED(b_{j-1}) \cup \{ a_i \} \). \( \square \)

Lemma 3.6. Let \( j \geq 2 \) and \( s(b_j) = a_i \). Then,

\[
ED(b_j) = \begin{cases} 
\text{null} & \text{if } i = 1 \text{ or } (a_{i-1}, b_{j-1}) \notin E, \\
ED(a_{i-1}) \cup \{ b_j \} & \text{if } i \neq 1 \text{ and } (a_{i-1}, b_{j-1}) \in E.
\end{cases}
\]

Proof. The proof of this lemma is similar to that of Lemma 3.5. \( \square \)

According to Lemmas 3.4, 3.5 and 3.6, if \( ED(a_m) \neq \textit{null} \) (resp. \( ED(b_n) \neq \textit{null} \)), then \( ED(a_m) \) (resp. \( ED(b_n) \)) can be computed by a greedy method as follows. Starting from \( a_m \) (resp. \( b_n \)), \( ED(a_m) \) (resp. \( ED(b_n) \)) repeatedly includes the largest non-dominated vertex from the opposite independent vertex set (i.e., \( A \) or \( B \)) until all vertices are
dominated. The details of computing \( ED(a_m) \) are described in Algorithm 1. Note that whether two vertices \( a_i \) and \( b_j \) are adjacent can be determined in constant time using the permutation diagram, i.e., \((a_i, b_j) \in E \) if and only if \((h-k)(\pi^{-1}(h) - \pi^{-1}(k)) < 0\), where \( h \) and \( k \) are the labels of \( a_i \) and \( b_j \), respectively. Hence, Algorithm 1 takes \( O(|A| + |B|) \) time. Similarly, \( ED(b_n) \) can be computed in \( O(|A| + |B|) \) time. Therefore, we have the following theorem.

**Theorem 3.1.** The weighted efficient domination problem can be solved in \( O(|V|) \) time for a bipartite permutation graph given with a permutation diagram.

### 4. An \( O(|V|) \) algorithm on distance-hereditary graphs

A graph is *distance-hereditary graph* if every two vertices have the same distance in every connected induced subgraph containing them. Many characterizations of distance-hereditary graphs were introduced in [1,20,24] and some algorithmic aspects concerning optimization problems were investigated in [6,10,14,15,20].

A vertex of \( G \) is *pendant* if its degree is one. Two vertices \( u, v \) of \( G \) form a *twin pair* if \( N(u) \setminus \{v\} = N(v) \setminus \{u\} \), and are called *false twins* if \((u, v) \notin E \) and *true twins* if \((u, v) \in E \). Denote by \( G[S] \) the subgraph induced by \( S \subseteq V \). A *one-vertex-extension (OVE) ordering* of \( G \) is an ordering \( v_1, v_2, \ldots, v_{|V|} \) of \( V \) such that for each \( 2 \leq i \leq |V| \), \( v_i \) is a pendant vertex or a twin of some other vertex in \( G[V_i] \), where \( V_i = \{v_1, v_2, \ldots, v_i\} \).

In [1], Bandelt and Mulder showed that \( G \) is a distance-hereditary graph if and only if \( G \) has an OVE ordering. Hammer and Maffray [20] used a pruning sequence to represent the same concept of OVE ordering. A *pruning sequence* is a sequence \((s_2, s_3, \ldots, s_{|V|})\) of words such that for each \( 2 \leq j \leq |V| \), the word \( s_j \) is \( v_jPv_i \), \( v_jFv_i \) or \( v_jTv_i \) for some \( i < j \) with the meaning that \( G[V_j] \) is obtained from \( G[V_{j-1}] \) by making \( v_j \) a pendant vertex adjacent to \( v_i \), a false twin of \( v_i \), or a true twin of \( v_i \), respectively. For example, \((2P1, 3P2, 4T1, 5F2, 6P4, 7P1)\) is a pruning sequence of the graph shown in Fig. 5(a). Hammer and Maffray gave an \( O(|V| + |E|) \) time algorithm that not only recognizes
Algorithm 1. The Computation of $ED(a_m)$.

$ED(a_m) = \emptyset$, $i = m$, $j = n$ and $stop = 0$;
while $stop = 0$ do
    while $(a_i, b_j) \in E$ do
        $j = j - 1$;
    endwhile
    $j = j + 1$; /* i.e., $b_j = s(a_i)$ */
    if $i = 1$ and $j = 1$ then
        $ED(a_m) = ED(a_m) \cup \{a_1\}$ and exit; /* By Lemma 3.3 */
    else if $i \neq 1$ and $j \neq 1$ and $(a_{i-1}, b_{j-1}) \in E$ then
        $ED(a_m) = ED(a_m) \cup \{a_i\}$; /* By Lemma 3.5 */
    else $ED(a_m) = null$ and exit; /* By Lemma 3.5 */
    endif
    $i = i - 1$ and $j = j - 1$;
endwhile

$i = i + 1$; /* i.e., $a_i = s(b_j)$ */
if $i = 1$ and $j = 1$ then
    $ED(a_m) = ED(a_m) \cup \{b_1\}$ and exit; /* By Lemma 3.3 */
else if $i \neq 1$ and $j \neq 1$ and $(a_{i-1}, b_{j-1}) \in E$ then
    $ED(a_m) = ED(a_m) \cup \{b_j\}$; /* By Lemma 3.6 */
else $ED(a_m) = null$ and exit; /* By Lemma 3.6 */
endif
$i = i - 1$ and $j = j - 1$;
endwhile

whether a given graph is distance-hereditary, but also generates a pruning sequence if it is [20].

Recently, Chang et al. introduced the concept of OVE tree based on an OVE ordering [11]. Given an OVE ordering $v_1, v_2, \ldots, v_{|V|}$ of $G$, the OVE tree of $G$, denoted by $ET(G)$, is defined as a rooted tree with root $v_1$ and constructed in the way as follows. Initially, $ET(G)$ has only root $v_1$. Next, nodes are added to $ET(G)$ from $v_2$ to $v_{|V|}$ in the way that for each $2 \leq j \leq |V|$, $v_j$ is the rightmost child of $v_i$ if $v_jPv_i$, $v_jFv_i$ or $v_jTv_i$. See Fig. 5(b) for an example. Clearly, $ET(G)$ can be constructed in $O(|V|)$ time if an OVE ordering of $G$ is given. We use $[v_i, v_j]$ to denote an edge in $ET(G)$, where $v_i$ is the parent of $v_j$. An edge $[v_i, v_j]$ is called a $P$ (resp. $F$ and $T$) edge if $v_jPv_i$ (resp. $v_jFv_i$ and $v_jTv_i$). If $G$ is connected, then edge $[v_1, v_2]$ in $ET(G)$ is either a $P$ or $T$ edge [1]. Without loss of generality, we assume that given distance-hereditary graph $G$ is connected and $[v_1, v_2]$ is a $P$ edge in $ET(G)$. Note that $G[V_i]$ is connected.
Before we go further, some terminology needs to be introduced. The subtree of ET(G) rooted at vi, denoted by ET(i), is the subtree of ET(G) induced by vi and all descendants of vi. Let V(i) be the set of all nodes in ET(i). The twin set of vi, denoted by TS(i), is the set of the nodes in ET(i) reachable from vi through only F or T edges. Note that vi ∈ TS(i). Suppose that vi has k children v_{c_1}, v_{c_2},..., v_{c_k} in ET(G). Then, for 1 ≤ j ≤ k, we use ET(i,c_j) to denote the subtree induced by vi, V(c_j), V(c_{j+1}),..., V(c_k).

Let V(i,c_j) be the set of all nodes in ET(i,c_j), TS(i,c_j) = TS(i) ∩ V(i,c_j), V_{R}(i,c_j) = V(i,c_j) \ V(c_j) and TS_{R}(i,c_j) = TS(i,c_j) \ V(c_j). For an edge [vi,v_j], we define ED(i,j) to be a minimum weighted efficient dominating (MWED) set of G[V(i,j)], ED_0(i,j) to be an MWED set of G[V(i,j)] satisfying the property that ED_0(i,j) ∩ TS(i,j) = ∅, ED_1(i,j) to be an MWED set of G[V(i,j)] satisfying the property that |ED_1(i,j) ∩ TS(i,j)| = 1, and ED̃(i,j) to be an MWED set of G[V(i,j) \ TS(i,j)] satisfying the property that no vertex in ED̃(i,j) is adjacent to any vertex in TS(i,j). If v_j is the leftmost child of vi, then for convenience, we use ED(i), ED_0(i), ED_1(i) and ED̃(i) to denote ED(i,j), ED_0(i,j), ED_1(i,j) and ED̃(i,j), respectively. Two disjoint subsets X,Y ⊆ V are said to form a join if every vertex in X is adjacent to all vertices in Y.

**Lemma 4.1.** Suppose that vi is a node in ET(G). Then, d(v_i,v_j) ≤ 2 for any v_j ∈ TS(i) with j ≠ i.

**Proof.** Note that i < j, and j > 2 since [v_1,v_2] is a P edge. Since G[V_{j-1}] is connected, there is a vertex v_h in G[V_{j-1}] with h ≠ i such that d(v_i,v_h) = 1. We then prove d(v_i,v_j) ≤ 2 and d(v_j,v_h) = 1 by induction on the level of node (i.e., the distance of node from v_j) in the connected subtree induced by TS(i). First, let the level of v_j be one. If [v_i,v_j] is a T edge, then d(v_i,v_j) = 1 and d(v_j,v_h) = 1. If [v_i,v_j] is an F edge, then d(v_j,v_h) = 1 and hence d(v_i,v_j) = 2. Next, let the level of v_j be k + 1 and w ∈ V_j be the parent of v_j. By the inductive hypothesis, we have d(v_i,w) ≤ 2 and d(w,v_h) = 1. Hence, d(v_j,v_h) = 1 and therefore d(v_j,v_i) ≤ 2. □

In the proof of Lemma 4.1, we can see that all vertices in TS(i) are adjacent to v_h. Hence, we have the following lemma immediately.

**Lemma 4.2.** Suppose that vi is a node in ET(G). Then, d(u,v) ≤ 2 for any two u,v ∈ TS(i).

Based on Lemma 4.2, we can conclude that ED(i) = min{ED_0(i), ED_1(i)} and ED(1) is an MWED set of G.

**Lemma 4.3** (Chang et al. [10]). Suppose that vi is a node in ET(G). Then, any vertex V(i) ∩ TS(i) is adjacent to only vertices in V(i).

**Lemma 4.4** (Chang et al. [10]). Suppose that [v_i,v_j] is a P or T edge in ET(G). Then, TS(j) and TS_R(i,j) form a join.
Lemma 4.5 (Chang et al. [10]). Suppose that \([v_i, v_j]\) is an F edge in \(ET(G)\). Then, every vertex in \(V(j)\) is not adjacent to any vertex in \(V_R(i, j)\), i.e., \(G[V(i, j)]\) is disconnected.

Lemma 4.6 (Chang et al. [10]). Suppose that \([v_i, v_j]\) is a P edge in \(ET(G)\). Then, every vertex in \(V(j)\) is adjacent to only vertices in \(V(j) \cup TS_R(i, j)\).

Lemma 4.7. Suppose that \([v_i, v_j]\) is an edge in \(ET(G)\). Then, no vertex in \(V(j)\) is adjacent to any vertex in \(V_R(i, j) \setminus TS_R(i, j)\).

Proof. Let \(u\) be any vertex in \(V_R(i, j) \setminus TS_R(i, j)\). Then, we distinguish the following two cases.

Case 1: There is a vertex \(v_k \in V_R(i, j) \setminus TS_R(i, j)\) such that \([v_i, v_k]\) is a P edge and \(u \in V(k)\). According to Lemma 4.6, \(u\) is adjacent to only vertex in \(V(k) \cup TS_R(i, k)\). Since \((V(k) \cup TS_R(i, k)) \cap V(j) = \emptyset\), \(u\) is not adjacent to any vertex in \(V(j)\).

Case 2: There is a vertex \(v_h \in TS_R(i, j)\) and \(v_k \in V(h)\) such that \([v_h, v_k]\) is a P edge and \(u \in V(k)\). According to Lemma 4.6, \(u\) is adjacent to only vertex in \(V(k) \cup TS_R(h, k)\). Since \((V(k) \cup TS_R(h, k)) \cap V(j) = \emptyset\), \(u\) is not adjacent to any vertex in \(V(j)\).

As mentioned above, no vertex in \(V(j)\) is adjacent to any vertex in \(V_R(i, j) \setminus TS_R(i, j)\).

The following lemma is clear from the definitions.

Lemma 4.8. Suppose that \(v_i\) is a leaf in \(ET(G)\). Then, \(TS(i) = \{v_i\}\), \(ED_0(i) = \emptyset\), \(ED_1(i) = \{v_i\}\) and \(\overline{ED}(i) = \emptyset\), where \(\emptyset\) means that such set does not exist.

Lemma 4.9. Suppose that \([v_i, v_j]\) is a P edge in \(ET(G)\) and \(v_j\) is the rightmost child of \(v_i\). Then, (1) \(TS(i, j) = \{v_i\}\), (2) \(ED_0(i, j) = ED_1(j)\), (3) \(ED_1(i, j) = \{v_i\} \cup \overline{ED}(j)\), and (4) \(\overline{ED}(i, j) = ED_0(j)\).

Proof. (1) It is clear that \(TS(i, j) = \{v_i\}\).

(2) Note that \(v_i \notin ED_0(i, j)\). Then, \(ED_0(i, j)\) needs to contain exactly one vertex in \(TS(j)\) to efficiently dominate \(v_i\) since all vertices in \(TS(j)\) are adjacent to \(v_i\) by Lemma 4.4 and no vertex in \(V(j) \setminus TS(j)\) is adjacent to \(v_i\) by Lemma 4.3. Hence, \(ED_0(i, j) = ED_1(j)\).

(3) Note that \(v_i \in ED_1(i, j)\). Hence, \(ED_1(i, j) = \{v_i\} \cup \overline{ED}(j)\) since all vertices in \(TS(j)\) are adjacent to \(v_i\) and no vertices in \(V(j) \setminus TS(j)\) is adjacent to \(v_i\).

(4) Clearly, \(\overline{ED}(i, j) = ED_0(j)\) since \(TS(i) = \{v_i\}\) and all vertices in \(TS(j)\) are adjacent to \(v_i\). □

Lemma 4.10. Suppose that \([v_i, v_j]\) is a T edge in \(ET(G)\) and \(v_j\) is the rightmost child of \(v_i\). Then, (1) \(TS(i, j) = \{v_i\} \cup TS(j)\), (2) \(ED_0(i, j) = \emptyset\), (3) \(ED_1(i, j) = \min\{\{v_i\} \cup \overline{ED}(j), ED_1(j)\}\), and (4) \(\overline{ED}(i, j) = ED_1(j)\).
Proof. Statement (1) is clear.

(2) Note that $ED_0(i, j)$ contains no vertex in $\{v_i\} \cup TS(j)$. By Lemma 4.3, no vertex in $V(j) \setminus TS(j)$ is adjacent to $v_i$. Suppose that $ED_0(i, j) \neq \text{null}$. Then, $v_i$ is not dominated by any vertex in $ED_0(i, j)$, a contradiction.

(3) Note that $ED_1(i, j)$ contains exactly one vertex in $\{v_i\} \cup TS(j)$. If $v_i \in ED_1(i, j)$, then $ED_1(i, j) = \{v_i\} \cup \overline{ED}(j)$ since all vertices in $TS(j)$ are adjacent to $v_i$ by Lemma 4.4 and no vertex in $V(j) \setminus TS(j)$ is adjacent to $v_i$ by Lemma 4.3. If $v_i \notin ED_1(i, j)$, then it is clear that $ED_1(i, j) = ED_1(j)$.

(4) Clearly, $\overline{ED}(i, j) = \overline{ED}(j)$ since $TS(i, j) = \{v_i\} \cup TS(j)$ and no vertex in $V(j) \setminus TS(j)$ is adjacent to $v_i$. □

Lemma 4.11. Suppose that $[v_i, v_j]$ is an $F$ edge in $ET(G)$ and $v_j$ is the rightmost child of $v_i$. Then, (1) $TS(i, j) = \{v_i\} \cup TS(j)$, (2) $ED_0(i, j) = \text{null}$, (3) $ED_1(i, j) = \{v_i\} \cup ED_0(i, j)$, and (4) $\overline{ED}(i, j) = \overline{ED}(j)$.

Proof. Statement (1) is clear.

(2) Note that $v_i \notin ED_0(i, j)$. By Lemma 4.5, no vertex in $V(j)$ is adjacent to $v_i$. Suppose that $ED_0(i, j) \neq \text{null}$. Then, $v_i$ is not dominated by any vertex in $ED_0(i, j)$, a contradiction.

(3) Note that $ED_1(i, j)$ contains exactly one vertex in $\{v_i\} \cup TS(j)$. Since no vertex in $V(j)$ is adjacent to $v_i$, $ED_1(i, j)$ needs to contain $v_i$ to efficiently dominate $v_i$. Hence, we have $ED_1(i, j) = \{v_i\} \cup ED_0(i, j)$.

(4) Clearly, $\overline{ED}(i, j) = \overline{ED}(j)$ since $TS(i, j) = \{v_i\} \cup TS(j)$ and no vertex in $V(j) \setminus TS(j)$ is adjacent to $v_i$. □

Lemma 4.12. Suppose that $[v_j, v_k]$ is a $P$ edge in $ET(G)$ and $v_k$ is the child of $v_j$ next to $v_j$. Then, (1) $TS(i, j) = TS(i, k)$, (2) $ED_0(i, j) = \min\{ED_0(j) \cup ED_0(i, k), ED_1(j) \cup \overline{ED}(i, k)\}$, (3) $ED_1(i, j) = \overline{ED}(j) \cup ED_1(i, k)$, and (4) $\overline{ED}(i, j) = ED_0(j) \cup \overline{ED}(i, k)$.

Proof. Statement (1) is clear.

(2) Note that $ED_0(i, j) \cap TS(i, k) = \emptyset$. By Lemma 4.3, every vertex in $V(j) \setminus TS(j)$ is adjacent to only vertices in $V(j)$. To efficiently dominate all vertices in $V(j) \setminus TS(j)$, either $ED_0(i, j) \cap TS(j) = \emptyset$ and hence $ED_0(i, j) = ED_0(j) \cup ED_0(i, k)$, or $|ED_0(i, j) \cap TS(j)| = 1$ and hence $ED_0(i, j) = ED_1(j) \cup \overline{ED}(i, k)$ since $TS(j)$ and $TS(i, k)$ form a join by Lemma 4.4 and no vertex in $V(j)$ is adjacent to any vertex in $V(i, k) \setminus TS(i, k)$ by Lemma 4.7.

(3) Note that $|ED_1(i, j) \cap TS(i, k)| = 1$. Hence, $ED_1(i, j) = \overline{ED}(j) \cup ED_1(i, k)$ since $TS(j)$ and $TS(i, k)$ form a join and no vertex in $V(j) \setminus TS(j)$ is adjacent to any vertex in $V(i, k)$.

(4) Note that $\overline{ED}(i, j) \cap TS(i, k) = \emptyset$, and $\overline{ED}(i, j) \cap TS(j) = \emptyset$ since $TS(j)$ and $TS(i, k)$ form a join. Hence, $\overline{ED}(i, j) = ED_0(j) \cup \overline{ED}(i, k)$ since no vertex in $V(j)$ is adjacent to any vertex $V(i, k) \setminus TS(i, k)$ by Lemma 4.7. □
Lemma 4.13. Suppose that \([v_i, v_j]\) is a \(T\) edge in \(ET(G)\) and \(v_k\) is the child of \(v_i\) next to \(v_j\). Then, (1) \(TS(i, j) = TS(j) \cup TS(i, k)\), (2) \(ED_0(i, j) = ED_0(j) \cup ED_0(i, k)\), (3) \(ED_1(i, j) = \min\{ED_1(j) \cup \overline{ED}(i, k), \overline{ED}(j) \cup ED_1(i, k)\}\), and (4) \(\overline{ED}(i, j) = \overline{ED}(j) \cup \overline{ED}(i, k)\).

Proof. Statement (1) is clear.

(2) Note that \(ED_0(i, j) \cap TS(j) = \emptyset\) and \(ED_0(i, j) \cap TS(i, k) = \emptyset\). Hence, \(ED_0(i, j) = ED_0(j) \cup ED_0(i, k)\) since no vertex in \(V(j) \setminus TS(j)\) is adjacent to any vertex in \(V(i, k)\) by Lemma 4.3 and no vertex in \(V(i, k) \setminus TS(i, k)\) is adjacent to any vertex in \(V(j)\) by Lemma 4.7.

(3) Note that \(|ED_1(i, j) \cap (TS(j) \cup TS(i, k))| = 1\). If \(|ED_1(i, j) \cap TS(j)| = 1\), then \(ED_1(i, j) = ED_1(j) \cup \overline{ED}(i, k)\) since \(TS(j)\) and \(TS(i, k)\) form a join by Lemma 4.4 and no vertex in \(V(j)\) is adjacent to any vertex in \(V(i, k) \setminus TS(i, k)\) by Lemma 4.7. If \(|ED_1(i, j) \cap TS(i, k)| = 1\), then \(ED_1(i, j) = \overline{ED}(j) \cup ED_1(i, k)\).

(4) Note that \(\overline{ED}(i, j) \cap (TS(j) \cup TS(i, k)) = \emptyset\). Hence, \(\overline{ED}(i, j) = \overline{ED}(j) \cup \overline{ED}(i, k)\) since no vertex in \(V(j) \setminus TS(j)\) is adjacent to any vertex in \(V(i, k)\) and no vertex in \(V(i, k) \setminus TS(i, k)\) is adjacent to any vertex in \(V(j)\).

Lemma 4.14. Suppose that \([v_i, v_j]\) is an \(F\) edge in \(ET(G)\) and \(v_k\) is the child of \(v_i\) next to \(v_j\). Then, (1) \(TS(i, j) = TS(j) \cup TS(i, k)\), (2) \(ED_0(i, j) = ED_0(j) \cup ED_0(i, k)\), (3) \(ED_1(i, j) = \min\{ED_1(j) \cup ED_0(i, k), ED_0(j) \cup ED_1(i, k)\}\), and (4) \(\overline{ED}(i, j) = \overline{ED}(j) \cup \overline{ED}(i, k)\).

Proof. By Lemma 4.5, no vertex in \(V(j)\) is adjacent to any vertex in \(V(i, k)\). Hence, this lemma holds.

Based on the lemmas above, we can design a dynamic programming algorithm to find an MWED set in a distance-hereditary graph \(G\). The details are described in Algorithm 2. Clearly, the time complexity of this algorithm is \(O(|V|)\).

Theorem 4.1. The weighted efficient domination problem can be solved in \(O(|V|)\) time for a distance-hereditary graph given with an OVE ordering.

Algorithm 2. Compute an MWED set on a distance-hereditary graph.

Input: An OVE ordering \(v_1, v_2, \ldots, v_{|V|}\) of a distance-hereditary graph \(G\).

Output: An MWED set \(D\) of \(G\).

Step 1: Construct the OVE tree \(ET(G)\) of \(G\);

for each leaf \(v_i\) in \(ET(G)\) do

\(ED_0(i) = \text{null}, ED_1(i) = \{v_i\}\) and \(\overline{ED}(i) = \emptyset\);
endfor

Step 2: for \(j = |V|\) to 2 do

\(\text{case 1: } s_j = v_jPv_i\) then
Compute $ED_0(i,j), ED_1(i,j)$ and $\overline{ED}(i,j)$ by Lemma 4.9 if $v_j$ is the rightmost child of $v_i$; otherwise, compute them by Lemma 4.12;

**case 2:** $s_j = v_j T v_i$

Compute $ED_0(i,j), ED_1(i,j)$ and $\overline{ED}(i,j)$ by Lemma 4.10 if $v_j$ is the rightmost child of $v_i$; otherwise, compute them by Lemma 4.13;

**case 3:** $s_j = v_j F v_i$

Compute $ED_0(i,j), ED_1(i,j)$ and $\overline{ED}(i,j)$ by Lemma 4.11 if $v_j$ is the rightmost child of $v_i$; otherwise, compute them by Lemma 4.14;

**endfor**

**Step 3:** $ED(1) = \min \{ ED_0(1), ED_1(1) \}$;

**if** $ED(1) \neq null$ **then** $D = ED(1)$; **else** $G$ has no efficient dominating set.

5. Conclusion

In this paper, we first showed that the efficient domination problem is NP-complete for planar bipartite graphs and chordal bipartite graphs. Finally, for the weighted efficient domination problem, we gave a greedy algorithm of $O(|V|)$ time for a bipartite permutation graph given with a permutation diagram and a dynamic programming algorithm of $O(|V|)$ time for a distance-hereditary graph given with an OVE ordering. Hence, the same problem is solvable in linear time for (6,2) chordal bipartite graphs and Ptolemaic graphs. It is worth mentioning that there is a bound between tractability and intractability of the weighted efficient domination problem for graph classes shown in Fig. 1. It would be of interest to know whether or not there is a polynomial time algorithm to solve the weighted efficient domination problem on other classes of graphs, such as convex bipartite graphs and strongly chordal graphs.

Acknowledgements

The authors would like to thank the anonymous referees for many constructive suggestions for the presentation of this paper.

References


