Discussiones Mathematicae
Graph Theory 43 (2023) 947–963
https://doi.org/10.7151/dmgt.2411

VERTEX-EDGE DOMINATION IN INTERVAL AND BIPARTITE PERMUTATION GRAPHS

Subhabrata Paul

Department of Mathematics
Indian Institute of Technology, Patna

e-mail: subhabrata@iitp.ac.in

Dinabandhu Pradhan[1]

Department of Mathematics & Computing
Indian Institute of Technology (ISM), Dhanbad

e-mail: dina@iitism.ac.in

AND

Shaily Verma

Department of Mathematics
Indian Institute of Technology, Delhi

e-mail: Shaily.Verma@maths.iitd.ac.in

Abstract

Given a graph $G = (V, E)$, a vertex $u \in V$ *ve-dominates* all edges incident to any vertex of $N_G[u]$. A set $D \subseteq V$ is a *vertex-edge dominating set* if, for any edge $e \in E$, there exists a vertex $u \in D$ such that u ve-dominates e. Given a graph G, our goal is to find a minimum cardinality ve-dominating set of G. In this paper, we designed two linear-time algorithms to find a minimum cardinality ve-dominating set for interval and bipartite permutation graphs.

Keywords: vertex-edge domination, linear time algorithm, interval graphs, bipartite permutation graphs.

2020 Mathematics Subject Classification: 05C69.

[1]Corresponding author.

1. Introduction

For a graph $G = (V, E)$, let $N_G(v)$ (or $N_G[v]$) be the *open* (respectively, *closed*) neighbourhood of v in G. A set $D \subseteq V$ is called a *dominating set* of a graph $G = (V, E)$ if $|N_G[v] \cap D| \geq 1$ for all $v \in V$. Domination problem is one of the classical problems in graph theory where the objective is to find a dominating set of minimum cardinality of a given graph. This minimum cardinality of a dominating set is known as the *domination number* of G and denoted by $\gamma(G)$. Over the last few decades, quite a few variants of the classical domination problem have been introduced and studied in the literature [6, 7]. In this paper, we have studied one variant of the domination problem, namely *vertex-edge domination* problem, also known as *ve-domination* problem.

Given a graph $G = (V, E)$, a vertex $u \in V$ *ve-dominates* all edges incident to any vertex of $N_G[u]$. A set $D \subseteq V$ is a *vertex-edge dominating set* (or simply a VED-set) if, for every edge $e \in E$, there exists a vertex $u \in D$ such that u ve-dominates e. The minimum cardinality among all the VED sets of G is called the *vertex-edge domination number* (or simply VED number) and is denoted by $\gamma_{ve}(G)$.

The vertex-edge domination problem was introduced by Peters [17] in his PhD thesis in 1986. However, it did not receive much attention until Lewis [14] in 2007 introduced some new parameters related to it and established many new results in his PhD thesis. In his PhD thesis, Lewis has given some lower bound on $\gamma_{ve}(G)$ for different graph classes like connected graphs, k-regular graphs, cubic graphs, etc. In [15], the authors have characterized the trees with equal domination and vertex-edge domination number. In [12], both upper and lower bounds on the ve-domination number of a tree have been proved. Some upper bounds on $\gamma_{ve}(G)$ and some relationship between the ve-domination number and other domination parameters have been proved in [3]. In [21], Żyliński has shown that for any connected graph G with $n \geq 6$, $\gamma_{ve}(G) \leq n/3$. On the algorithmic side, Lewis has proved that the ve-domination problem is NP-complete for bipartite, chordal, planar, and circle graphs. Approximation algorithm and approximation hardness results are proved in [14]. In [16], the authors designed a linear-time algorithm for the ve-domination problem in block graphs and showed that this problem remains NP-complete for undirected path graphs. Recently, Jena and Das [9] have studied the ve-domination problem in unit disk graphs. Other variations of the ve-domination problem have also been studied in literature [2, 4, 5, 10, 11, 20].

The main focus of this article is to study the polynomial solvability of the ve-domination problem in subclasses of chordal and bipartite graphs. We have proved that this problem can be solved in linear-time for interval graphs, an important subclass of chordal graphs. We have also designed a linear-time algorithm to solve the ve-domination problem for bipartite permutation graphs, an important subclass of bipartite graphs.

The rest of the paper is organized as follows. Section 2 deals with some pertinent definitions and notations that are used in the subsequent sections. In Section 3, we designed a linear-time algorithm based on greedy technique, to solve the ve-domination problem for interval graphs. In Section 4, using dynamic programming approach, we designed another linear-time algorithm to solve the ve-domination problem for bipartite permutation graphs. Finally, Section 5 concludes the paper.

2. Definitions and Notations

A graph G is a *chordal graph* if every cycle in G of length at least 4 has a *chord* i.e., an edge joining two non-consecutive vertices of the cycle. Let $\mathscr{F}$ be a nonempty family of sets. A graph $G = (V, E)$ is called an *intersection graph* for a finite family $\mathscr{F}$ of a nonempty set if there is a one-to-one correspondence between $\mathscr{F}$ and V such that two sets in $\mathscr{F}$ have nonempty intersection if and only if their corresponding vertices in V are adjacent. We call $\mathscr{F}$ an *intersection model* of G. For an intersection model $\mathscr{F}$, we use $G(\mathscr{F})$ to denote the intersection graph for $\mathscr{F}$. If $\mathscr{F}$ is a family of intervals on a real line, then $G(\mathscr{F})$ is called an *interval graph* for $\mathscr{F}$ and $\mathscr{F}$ is called an *interval model* of $G(\mathscr{F})$. An $O(n + m)$ time algorithm has been given in [1] for recognizing an interval graph and constructing an interval model using PQ-trees.

We use the standard notations $[k] = \{1, 2, \ldots, k\}$ and $[k, k'] = \{k, k + 1, \ldots, k'\}$ for $k < k'$. Suppose G is an interval graph and I is its interval model. For every vertex $v_i \in V$, let I_i be the corresponding interval, and let a_i and b_i denote the left endpoint and right endpoint of the interval I_i, respectively. We order the vertices of G as $\sigma = (v_1, v_2, \ldots, v_n)$ in increasing order of their right endpoints. It is easy to see that if $v_iv_k \in E$ with $i < k$, then $v_jv_k \in E$ for every $j \in [i + 1, k]$. We call such an ordering of G as an *interval ordering.* The interval ordering can be computed from the set of maximal cliques of a given interval graph $G = (V, E)$ in linear-time (see [18]). Let $V_i = \{v_i, v_{i+1}, \ldots, v_n\}$ and $G_i = G[V_i]$ for $i \in [n]$. If G is a connected interval graph, then G_i is also connected. For the sake of simplicity, if not specified, we consider only connected interval graphs.

A vertex v of a graph G is called a *simplicial vertex* if $N_G[v]$ induces a complete subgraph (a clique) of G. An ordering $\sigma = (v_1, v_2, \ldots, v_n)$ of vertices of a graph G is called a perfect elimination ordering, abbreviated PEO, of G if for every $i \in [n]$, v_i is simplicial in $G[V_i]$.

The following observation can immediately be deduced by the virtue of an interval ordering of a connected interval graph.

Observation 1. *If G is a connected interval graph with an interval ordering $\sigma = (v_1, v_2, \ldots, v_n)$, then the following are true.*

(a) *If* $v_iv_k \in E(G)$ *with* $i < k$, *then* $v_iv_j \in E(G)$ *for every* $i < j < k$.

(b) σ *is a* PEO *of* G *i.e., for every* $i \in [n]$, v_i *is simplicial in* G_i.

Observation 2. *If* $\sigma = (v_1, v_2, \ldots, v_n)$ *is an interval ordering of a connected interval graph* G, *then the following are true.*

(a) *If* $k \leq i$, *then* $N_G[v_k] \cap V(G_i) \subseteq N_{G_i}[v_i]$.

(b) *If* $v_k \in N_{G_i}[v_i]$, *then* $N_{G_i}[v_i] \subseteq N_{G_i}[v_k]$.

(c) *For every* i, j, *and* k *with* $v_j, v_k \in N_{G_i}[v_i]$, $N_{G_i}[v_j] \subseteq N_{G_i}[v_k]$.

Proof. (a) If $k = i$, then we are done. So assume that $k < i$ and let $v_r \in N_G[v_k] \cap V(G_i)$. By Observation 1(a), $v_rv_i \in E(G)$. So $v_r \in N_G[v_i]$ and hence $v_r \in N_{G_i}[v_i]$, completing the proof of (a).

(b) If $k = i$, then we are done. So assume that $k \neq i$ and let $v_r \in N_{G_i}[v_i]$. Since σ is an interval ordering of G, v_i is a simplicial vertex in G_i. This implies that $v_r \in N_G[v_k]$ and hence $v_r \in N_{G_i}[v_k]$, consequently, $N_{G_i}[v_i] \subseteq N_{G_i}[v_k]$. This completes the proof of (b).

(c) Let $v_r \in N_{G_i}[v_j]$. If $i = r$, then by (b), it is clear that $v_i \in N_{G_i}[v_i] \subseteq N_{G_i}[v_k]$. So assume that $i < r$. If $r = j$ or $r = k$, then as $i < j < k$ and by Observation 1(a), $v_jv_k \in E(G)$ and hence we are done. Next assume that $r \neq j$ and $r \neq k$. This implies that either $r < k$ or $k < r$. If $r < k$, then $i < r < k$ and hence by Observation 1(a), $v_rv_k \in E(G)$; thus $v_r \in N_{G_i}[v_j] \subseteq N_{G_i}[v_k]$. If $r > k$, then $j < k < r$. So by Observation 1(a), $v_kv_r \in E(G)$; thus $v_r \in N_{G_i}[v_j] \subseteq N_{G_i}[v_k]$. This completes the proof of (c). ∎

A graph $G = (V, E)$ is called a *permutation graph* if there exists a one-to-one correspondence between $V(G)$ and a set of line segments between two parallel lines such that two vertices are adjacent if and only if their corresponding line segments intersect. If $G = (V, E)$ is both a bipartite graph and a permutation graph, then it is called a *bipartite permutation graph*. Figure 1 illustrates the correspondence between the vertices of a bipartite permutation graph and a set of line segments between two parallel lines. The bipartite permutation graph was studied by Spinrad *et al.* [19] and also by Lai and Wei [13].

Let $\beta = (y_1, y_2, \ldots, y_{n_2})$ be some ordering of Y of a bipartite graph $G = (X, Y, E)$. A subset of Y is called a *segment* of Y if its elements are consecutive in β. The ordering β has the *convex property* if, for each vertex $x \in X$, $N_G(x)$ is a segment in β. An ordering of Y with the convex property is called a *convex ordering*. A bipartite graph $G = (X, Y, E)$ is said to be *convex* on Y if there exists a convex ordering of Y. The term convex on X is defined similarly.

Let $\sigma = (x_1, x_2, \ldots, x_{n_1}, y_1, y_2, \ldots, y_{n_2})$ be an ordering of $X \cup Y$ of $G = (X, Y, E)$ such that $\alpha = (x_1, x_2, \ldots, x_{n_1})$ is an ordering of X and $\beta = (y_1, y_2, \ldots, y_{n_2})$ is an ordering of Y. For any segment $S = \{s_a, s_{a+1}, \ldots, s_b\}$ in X (or Y), define first(S) and last(S), respectively, to be the index of the first element in S

and that of the last element in S with respect to the ordering induced on S. For two segments A and B in the same vertex set, define $A \preceq B$ if $\text{first}(A) \leq \text{first}(B)$ and $\text{last}(A) \leq \text{last}(B)$.

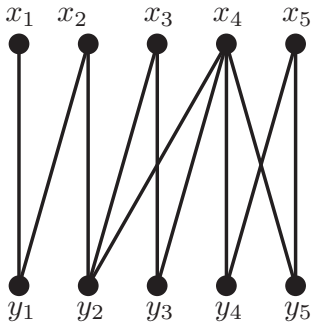

Figure 1. An example of a bipartite permutation graph.

An ordering $\sigma = (x_1, x_2, \ldots, x_{n_1}, y_1, y_2, \ldots, y_{n_2})$ of $X \cup Y$ is called a *forward-convex ordering* if $(y_1, y_2, \ldots, y_{n_2})$ is a convex ordering of Y and for every pair of vertices $x_i, x_j \in X$ with $i < j$, $N_G(x_i) \preceq N_G(x_j)$. The graph G is said to be *forward-convex* if there exists a forward-convex ordering of G. The graph shown in Figure 1 is also a forward-convex graph as $(x_1, x_2, x_3, x_4, x_5, y_1, y_2, y_3, y_4, y_5)$ is a forward-convex ordering. A linear-time algorithm for computing forward-convex ordering of a bipartite permutation graph is presented in [13].

The bipartite permutation graphs are same as the forward-convex graphs as can be seen from the following theorem.

Theorem 1 [13]**.** *The following statements are equivalent for a bipartite graph* $G = (X, Y, E)$*.*

(a) *G is a bipartite permutation graph.*

(b) *G is forward-convex.*

By Theorem 1, it is clear that bipartite permutation graphs have forward-convex orderings.

For notational convenience, for a given set $X = \{x_1, x_2, \ldots, x_{n_1}\}$, we denote X_i as the set $\{x_i, x_{i+1}, \ldots, x_{n_1}\}$ for every $i \in [n_1]$. Similarly given $Y = \{y_1, y_2, \ldots, y_{n_2}\}$, we denote Y_i as the set $\{y_i, y_{i+1}, \ldots, y_{n_2}\}$ for every $i \in [n_2]$.

Let $\sigma = (x_1, x_2, \ldots, x_{n_1}, y_1, y_2, \ldots, y_{n_2})$ of the vertices of a bipartite graph $G = (X, Y, E)$. Let $N_k(x) = N_G(x) \cap Y_k$, for a vertex $x \in X$. Similarly, assume that $N_k(y) = N_G(y) \cap X_k$, for a vertex $y \in Y$.

Lemma 1 [8]**.** *If $\sigma = (x_1, x_2, \ldots, x_{n_1}, y_1, y_2, \ldots, y_{n_2})$ is a forward-convex ordering of a connected bipartite permutation graph $G = (X, Y, E)$, then the following are true.*

(a) *If* $y_j, y_k \in N_G(x_i)$ *with* $j < k$, *then* $N_i(y_j) \subseteq N_i(y_k)$.

(b) *If* $x_j, x_k \in N_G(y_i)$ *with* $j < k$, *then* $N_i(x_j) \subseteq N_i(x_k)$.

Lemma 2. *Let* $G = (X, Y, E)$ *be a connected bipartite permutation graph with forward-convex ordering* $\sigma = (x_1, x_2, \ldots, x_{n_1}, y_1, y_2, \ldots, y_{n_2})$. *If* $\ell = \text{last}(N_G(x_1))$, $k = \text{last}(N_G(y_\ell))$, $\ell' = \text{first}(N_G(x_{k+1}))$, $r = \text{last}(N_G(y_1))$, $s = \text{last}(N_G(x_r))$, *and* $r' = \text{first}(N_G(y_{s+1}))$, *then the following hold.*

(a) $x_1y_1 \in E$ *and* $x_{n_1}y_{n_2} \in E$.

(b) *The vertices of* $\{y_j : j \in [\ell+1, \ell'-1]\}$ *are isolated vertices in* $G[X_{k+1} \cup Y_{\ell+1}]$.

(c) *The vertices of* $\{x_i : i \in [r+1, r'-1]\}$ *are isolated vertices in* $G[X_{r+1} \cup Y_{s+1}]$.

(d) *The graphs* $G' = G[X_{k+1} \cup Y_{\ell'}]$ *and* $G'' = G[X_{r'} \cup Y_{s+1}]$ *are connected bipartite permutation graphs.*

Proof. Let $x_1y_1 \notin E$. Note that x_1 and y_1 are not isolated vertices as G is a connected graph. Suppose that $a = \text{first}(N_G(x_1))$ and $b = \text{first}(N_G(y_1))$, that is, $x_1y_a, x_by_1 \in E(G)$. Since $x_1y_1 \notin E(G)$, we have $a > 1$ and $b > 1$. Moreover, $\text{first}(N_G(x_1)) \leq \text{first}(N_G(x_b))$, due to the forward-convex ordering σ of G, which is not possible as $\text{first}(N_G(x_b)) = 1$ and $\text{first}(N_G(x_1)) \neq 1$. Thus we get a contradiction. Hence $x_1y_1 \in E$. By a similar argument, we can deduce that $x_{n_1}y_{n_2} \in E$. Thus the property (a) is true.

Let $A = \{y_j : j \in [\ell + 1, \ell' - 1]\}$ and $B = \{x_i : i \in [r + 1, r' - 1]\}$. Since $y_{\ell'}$ is the minimum indexed neighbour of x_{k+1}, any vertex in the set A is not adjacent to x_{k+1}. Now, due to the forward-convex ordering σ, $\text{first}(N_G(x_{k+1})) \leq \text{first}(N_G(x_a))$ for every $a \in [k + 2, n_1]$. It implies that for every vertex y_j of A, $\text{last}(N_G(y_j)) \leq k$. Therefore, every vertex of A is an isolated vertex in the graph $G[X_{k+1} \cup Y_{\ell+1}]$. Hence (b) is true. Similarly, the property (c) can be proved.

Next, we prove the property (d). Let $G' = G[X_{k+1} \cup Y_{\ell'}]$. Since $\sigma = (x_1, x_2, \ldots, x_{n_1}, y_1, y_2, \ldots, y_{n_2})$ is a forward-convex ordering of G, $\sigma' = (x_{k+1}, x_{k+2}, \ldots, x_{n_1}, y_{\ell'}, y_{\ell'+1}, \ldots, y_{n_2})$ is a forward-convex ordering of the graph G'. Therefore, G' is a connected bipartite permutation graph as the graph G' admits a forward-convex ordering and $y_{\ell'}x_{k+1} \in E(G')$. By a similar argument, we can say that the graph $G'' = G[X_{r'} \cup Y_{s+1}]$ is also a connected bipartite permutation graph. Hence, the property (d) holds. ∎

3. Interval Graphs

In this section, we design a linear-time algorithm for finding a minimum VED-set of a given connected interval graph G. The algorithm is based on greedy technique where at each iteration, we choose a vertex in the VED-set which ve-dominates more number of edges.

3.1. The algorithm

First we give a comprehensive description of the proposed algorithm. Given a connected interval graph $G = (V, E)$, let $\sigma = (v_1, v_2, \ldots, v_n)$ be an interval ordering of V. For $i, j \in [n]$ with $i < j$, we define the following notations: $V_{[i,j]} = \{v_i, v_{i+1}, \ldots, v_j\}$, $V_i = V_{[i,n]}$, $G_i = G[V_i]$ and $i^* = \max\{k : v_k \in N_{G_i}[v_i]\}$.

Algorithm 1: VEDS-INTERVAL(G)

Input: A connected interval graph $G = (V, E)$;
Output: A minimum VED-set D of G;

Obtain an interval ordering $\sigma = (v_1, v_2, \ldots, v_n)$ of G;
$S = \emptyset$;
$i = 1$;
$D[v_i] = 0$ for all $i \in [n]$;
while $(i \leq n - 1)$ **do**
  if (*there is an edge incident on v_i that is not ve-dominated*) **then**
    $C(v_i) = \{v_\ell : \ell \in [i, i^*]$ and $D[v_\ell] = 0\}$;
    For every $v_j \in C(v_i)$, let $j^+ = \min\{\ell : \ell \in [j + 1, i^*], D[v_\ell] = 0$ and $v_\ell \in N_G[v_j]\}$;
    Let $C^+(v_i) = \{v_{j^+} : v_j \in C(v_i)\}$;
    Let $\rho(v_i) = \min\{\ell^* : v_\ell \in C^+(v_i)\}$;
    $S = S \cup \{v_{\rho(v_i)}\}$;
    $D[v] = 1$ for every $v \in N_G[v_{\rho(v_i)}]$;
    $i = \rho(v_i) + 1$;
  else
    $i = i + 1$;
return S;

We process the vertices of G according to the interval ordering $\sigma = (v_1, v_2, \ldots, v_n)$ of G and construct a VED-set, say S, of G at the termination of our algorithm. At any iteration, while processing a vertex v_i, a new vertex is selected into the set S if all the edges incident on v_i are not ve-dominated. To track whether the edges are ve-dominated or not, we use a label D on the vertices, rather than edges, of G. Initially, $D[v] = 0$ for all $v \in V(G)$. If a vertex x is selected into S, then $D[v]$ is made 1 for every $v \in N_G[x]$. In other words, $D[v]$ indicates whether all the edges incident on v are ve-dominated by the so far constructed set S or not. Note that, an edge xy is not ve-dominated by the so far constructed set S if $D[x] = D[y] = 0$. At any iteration, while processing a vertex v_i, we first check whether all the edges incident on v_i are ve-dominated or not by the so far constructed set S. If all the edges incident on v_i are ve-dominated, then we do

not include any new vertex to the set S. Otherwise, we find a suitable vertex to ve-dominate the incident edges on v_i. In fact, we choose the maximum indexed neighbour of the vertex v_{j^+}, where $v_j v_{j^+}$ is an edge which is not ve-dominated by the so far constructed set S, $j \in [i, i^*]$, and j^+ is the minimum index over all such edges. The details are provided in the algorithm.

3.2. Correctness of VEDS-Interval(G)

Next, we show that VEDS-Interval(G) outputs a minimum VED-set of a given connected interval graph. Suppose that VEDS-Interval(G) executes for k number of iterations. Then $k \leq n$. Let D_r, $r \in [k]$ be the set constructed by VEDS-Interval(G) after the execution of the r-th iteration. Clearly $D_0 = \emptyset$. First, we show that, while processing a vertex v_i, by selecting a new vertex into VED-set, we are not only ve-dominating all the edges incident on v_i but also we are ve-dominating all the edges incident on many other vertices.

Lemma 3. *Suppose v_i is processed at the r-th iteration of the algorithm for some $r \in [k]$. If $v_{\rho(v_i)}$ is chosen by the algorithm, then all the edges incident on the vertices of $V_{[i,\rho(v_i)]}$ are ve-dominated by $D_{r-1} \cup \{v_{\rho(v_i)}\}$.*

Proof. Let $v_d \in C^+(v_i)$ such that $d^* = \rho(v_i)$ and $v_c \in C(v_i)$ such that $d = c^+$. Clearly $d \in [i, \rho(v_i)]$ and $\rho(v_i) \geq i^*$. Let $v_a \in V_{[i,\rho(v_i)]}$ be arbitrary. If $a \geq d$, then since σ is an interval ordering of G, by Observation 1(a), $v_a v_{\rho(v_i)} \in E(G)$. This implies that every edge incident on v_a is ve-dominated by $v_{\rho(v_i)}$. Now assume that $a < d$. If $N_G[v_a] \cap D_{r-1} \neq \emptyset$, then every edge incident on v_a is ve-dominated by $D_{r-1} \cup \{v_{\rho(v_i)}\}$. So we may assume that $N_G[v_a] \cap D_{r-1} = \emptyset$. Let $v_a v_b$ be an edge incident on v_a. If $b \geq d$, then since σ is an interval ordering of G, by Observation 1(a), $v_b v_{\rho(v_i)} \in E(G)$. This implies that $v_a v_b$ is ve-dominated by $v_{\rho(v_i)}$. If $b \in [i, d-1]$, then by the choice of d, $N_G[v_b] \cap D_{r-1} \neq \emptyset$. This implies that $v_a v_b$ is ve-dominated by D_{r-1}. So all the edges incident on the vertices of $V_{[i,\rho(v_i)]}$ are ve-dominated by $D_{r-1} \cup \{v_{\rho(v_i)}\}$, completing the proof of the lemma. ■

By Lemma 3, the following can be observed for the algorithm VEDS-Interval(G).

Observation 3. *For every $r \in [k]$, at the beginning of the r-th iteration of the algorithm, all the edges incident on the vertices of $V_{[1,i-1]}$ are ve-dominated by D_{r-1}, where v_i is the vertex considered at the i-th iteration of the algorithm.*

Next lemma shows that at each iteration algorithm VEDS-Interval(G) adds a new vertex into VED-set in such a way that the minimality is maintained.

Lemma 4. *Suppose that v_i is processed at the r-th iteration of the algorithm for some $r \in [k]$ and there is an edge incident on v_i that is not ve-dominated by D_{r-1}.*

If D_{r-1} is contained in some minimum VED*-set of G, then there is a minimum* VED*-set of G containing $D_{r-1} \cup \{v_{\rho(v_i)}\}$.*

Proof. Let D be a minimum VED-set of G such that $D_{r-1} \subseteq D$. By Observation 3, all the edges incident on the vertices of $V_{[1,r-1]}$ are ve-dominated by D_{r-1}. Since there is an edge incident on v_i that is not ve-dominated by D_{r-1}, we have $C(v_i) \neq \emptyset$ and $C^+(v_i) \neq \emptyset$; thus $\rho(v_i)$ exists. Let $v_d \in C^+(v_i)$ such that $d^* = \rho(v_i)$ and $v_c \in C(v_i)$ such that $d = c^+$. Clearly $\rho(v_i) \geq i^*$ and $c < d$.

Let $v_s \in D$ such that $v_c v_d$ is ve-dominated by v_s. If $v_{\rho(v_i)} \in D$, then we are done. So we may assume that $v_{\rho(v_i)} \notin D$. So it is clear that $s \neq \rho(v_i)$. Notice that $v_s \in N_G[v_c] \cup N_G[v_d]$. To proceed further, we prove the following claims.

Claim 1. *If $s = c$, then D can be modified to D' such that $D_{r-1} \cup \{v_{\rho(v_i)}\} \subseteq D'$.*

Proof. To prove this, we show that all the edges ve-dominated by v_c are also ve-dominated by $D_{r-1} \cup \{v_{\rho(v_i)}\}$. By Observation 3, all the edges having at least one endpoint appearing before i with respect to σ are ve-dominated by D_{r-1}. So we need to prove that all edges, whose both endpoints belong to $V_{[i,n]}$ and are ve-dominated by v_c, are also ve-dominated by $D_{r-1} \cup \{v_{\rho(v_i)}\}$. By Lemma 3, all the edges incident on v_c are ve-dominated by $D_{r-1} \cup \{v_{\rho(v_i)}\}$.

Claim 1.1. *Every edge of G_i incident on some vertex of $N_{G_i}(v_c)$ is ve-dominated by $D_{r-1} \cup \{v_{\rho(v_i)}\}$.*

Proof. Let $v_a \in N_{G_i}(v_c)$ be arbitrary and $v_a v_b$ be an edge of G_i incident on v_a. If $a \leq \rho(v_i)$, then by Lemma 3, all edges incident on v_a are ve-dominated by $D_{r-1} \cup \{v_{\rho(v_i)}\}$. So we may assume that $a > \rho(v_i)$. Then $v_d, v_a \in N_{G_c}[v_c]$. By Observation 2(c), $N_{G_c}[v_d] \subseteq N_{G_c}[v_a]$ and thus $v_{\rho(v_i)} \in N_{G_c}[v_a]$. This implies that the edges incident on v_a are ve-dominated by $v_{\rho(v_i)}$. This completes the proof of Claim 1.1. □

Let $D' = (D \setminus \{v_c\}) \cup \{v_{\rho(v_i)}\}$. By Observation 3 and Claim 1.1, all edges incident on some vertex of $N_G[v_c]$ are ve-dominated by $D_{r-1} \cup \{v_{\rho(v_i)}\}$. So D' is a minimum VED-set of G such that $D_{r-1} \subseteq D'$, completing the proof of Claim 1. □

Claim 2. *If $s < c$, then D can be modified to D' such that $D_{r-1} \cup \{v_{\rho(v_i)}\} \subseteq D'$.*

Proof. Let $D' = (D \setminus \{v_s\}) \cup \{v_{\rho(v_i)}\}$. By Observation 3 and Lemma 3, every edge incident on v_s is ve-dominated by $D_{r-1} \cup \{v_{\rho(v_i)}\}$. Let $v_a \in N_G(v_s)$ be arbitrary. If $a < i$, then by Observation 3, every edge incident on v_a is ve-dominated by D_{r-1} and hence by $D_{r-1} \cup \{v_{\rho(v_i)}\}$. If $a \in [i, c]$, then by Lemma 3, every edge incident on v_a is ve-dominated by $D_{r-1} \cup \{v_{\rho(v_i)}\}$. If $a > c$, then Observation 2(a) implies that $v_a \in N_G[v_s] \cap V_c \subseteq N_{G_c}[v_c]$. So by Claim 1.1, all edges incident

on some vertex of $N_G(v_s) \cap V_c$ are ve-dominated by $D_{r-1} \cup \{v_{\rho(v_i)}\}$. Hence D' is a minimum VED-set of G such that $D_{r-1} \cup \{v_{\rho(v_i)}\} \subseteq D'$, completing the proof of Claim 2. □

Claim 3. *If $s > c$, then D can be modified to D' such that $D_i \subseteq D'$.*

Proof. Since $v_s \in N_G[v_c] \cup N[v_d]$ and $s > c$, we have $v_s \in N_{G_c}[v_c] \cup N_{G_c}[v_d]$. Moreover, by Observation 2(b), $N_{G_c}[v_c] \subseteq N_{G_c}[v_d]$. Thus $v_s \in N_{G_c}[v_d]$.

Claim 3.1. *Every edge incident on a vertex of $N_{G_i}[v_d]$ is ve-dominated by $D_{r-1} \cup \{v_{\rho(v_i)}\}$.*

Proof. Since $d^* = \rho(v_i)$, every edge incident on v_d is ve-dominated by $v_{\rho(v_i)}$. Let $v_a \in N_{G_i}(v_d)$ be arbitrary. Then $a \leq \rho(v_i)$. So by Lemma 3, every edge incident on v_a is ve-dominated by $D_{r-1} \cup \{v_{\rho(v_i)}\}$. This completes the proof of Claim 3.1. □

Claim 3.2. *Every edge incident on a vertex of $N_{G_i}[v_s]$ is ve-dominated by $D_{r-1} \cup \{v_{\rho(v_i)}\}$.*

Proof. Recall that $v_s \in N_{G_c}[v_d]$. If $s = d$, then by Claim 3.1, every edge incident on a vertex of $N_{G_i}[v_d]$ is ve-dominated by $D_{r-1} \cup \{v_{\rho(v_i)}\}$. If $s \neq d$, then $s \leq \rho(v_i)$. So by Lemma 3, every edge incident on v_s is ve-dominated by $D_{r-1} \cup \{v_{\rho(v_i)}\}$. Let $v_a \in N_{G_i}(v_s)$ be arbitrary. If $a \in [i, \rho(v_i)]$, then by Lemma 3, every edge incident on v_a is ve-dominated by $D_{r-1} \cup \{v_{\rho(v_i)}\}$. If $a > \rho(v_i)$, then by Observation 1(a), $v_a v_{\rho(v_i)} \in E(G)$. This implies that every edge incident on v_a is ve-dominated by $v_{\rho(v_i)}$. This completes the proof of Claim 3.2. □

Let $D' = (D \setminus \{v_s\}) \cup \{v_{\rho(v_i)}\}$. By Observation 3, Claim 1.1, and Claim 3.2, every edge incident on a vertex of $N_G[v_s]$ is ve-dominated by $D_{r-1} \cup \{v_{\rho(v_i)}\}$. Hence D' is a minimum VED-set of G such that $D_{r-1} \cup \{v_{\rho(v_i)}\} \subseteq D'$, completing the proof of Claim 3. □

We now return to the proof of Lemma 4. By Claims 1–3, we conclude that there is a minimum VED-set of G containing $D_{r-1} \cup \{v_{\rho(v_i)}\}$. This completes the proof of Lemma 4. ■

By Observation 3, notice that at the end of the algorithm, D_k is a VED-set of G. We next prove that D_k is a minimum VED-set of G. We use induction on r (number of iterations) that for every $r \in [k] \cup \{0\}$, D_r is contained in some minimum VED-set of G. Note that $D_0 = \emptyset$, for $r = 0$ and hence it is contained in some minimum VED-set of G. Now assume that D_{r-1} is contained in some minimum VED-set, say D of G. Let $\sigma = (v_1, v_2, \ldots, v_n)$ be an interval ordering of G and v_i be the vertex that is considered at the r-th iteration of the algorithm.

If every edge incident on v_i is ve-dominated by D_{r-1} (i.e., $D[v_i] = 1$ or $D[v] = 1$ for all $v \in N_G(v_i)$), then the algorithm does not select any vertex. So $D_r = D_{r-1}$ and hence D_r is contained in D. If there is an edge incident on v_i that is not ve-dominated by D_{r-1} (i.e., $D[v_i] = 0$ and $D[v] = 0$ for some $v \in N_G(v_i)$), then the algorithm selects the vertex $v_{\rho(v_i)}$. So $D_r = D_{r-1} \cup \{v_{\rho(v_i)}\}$. By Lemma 4, there is a minimum VED-set containing $D_r = D_{r-1} \cup \{v_{\rho(v_i)}\}$. So by induction, we have the following lemma.

Lemma 5. *Algorithm* VEDS-INTERVAL(G) *outputs a minimum* VED*-set of a given connected interval graph G.*

3.3. Running time of VEDS-INTERVAL(G)

Let G be a connected interval graph having n vertices and m edges. The correctness of the algorithm VEDS-INTERVAL(G) follows from Lemma 5. We now discuss the running time of the algorithm VEDS-INTERVAL(G). An interval ordering of an interval graph can be constructed in $O(n + m)$ time [18]. The running time of the algorithm VEDS-INTERVAL(G) depends on the execution of the "**while**" loop. Assume that v_i is considered at the r-th iteration of the algorithm VEDS-INTERVAL(G). We maintain an array D on the vertices. Initially, $D[v] = 0$ for every $v \in V(G)$. Once a vertex u is selected by the algorithm, $D[v]$ is made 1 for every $v \in N_G[u]$. The "**while**" loop depends on checking the condition "there is an edge incident on v_i that is not ve-dominated", computing the sets $C(v_i)$ and $C^+(v_i)$, finding the index $\rho(v_i)$, and updating the label D on the vertices of $N_G[v_{\rho(v_i)}]$. Notice if an edge xy is ve-dominated by the so far constructed set S, then $S \cap (N_G[x] \cup N[y]) \neq \emptyset$; otherwise $S \cap (N_G[x] \cup N[y]) = \emptyset$. So at the r-th iteration if $D[v_i] = 1$ or $D[v] = 1$ for all $v \in N_G(v_i)$, then we conclude that all the edges incident on v_i are ve-dominated by the so far constructed set S; otherwise there is an edge incident on v_i that is not ve-dominated by the so far constructed set S. This can be checked in at most $O(d_G(v_i))$ time. If $D[v_i] = 1$ or $D[v] = 1$ for all $v \in N_G(v_i)$, then the algorithm moves to $(r + 1)$-th iteration, concluding that the "**while**" loop can be executed in $O(d_G(v_i))$ time. If $D[v_i] = 0$ and $D[v] = 0$ for some $v \in N_G(v_i)$ (in fact $v \in N_{G_i}(v_i)$), then the algorithm computes $C(v_i), C^+(v_i)$ and $\rho(v_i)$. Notice that $C(v_i)$ can be computed in at most $O(d_G(v_{i^*}))$ time. Once $C(v_i)$ is computed, v_{j^+} can be computed in at most $d_G(v_j)$ time. So the set $C^+(v_i)$ can be computed in $O\left(\sum_{v \in C(v_i)} d_G(v)\right)$ time. Then $\rho(v_i)$ can be computed in $O(|C^+(v_i)|)$ time. The update of label D for the vertices of $N_G[v_{\rho(v_i)}]$ takes at most $O(d_G(v_{\rho(v_i)}))$ time. So at the r-th iteration, the "**while**" loop takes at most

$$O(d_G(v_i)) + O\left(\sum_{v \in C(v_i)} d_G(v)\right) + O(|C^+(v_i)|) + O(d_G(v_{\rho(v_i)})) \text{ time.}$$

Since $|C^+(v_i)| \leq |C(v_i)|$ and $|C(v_i)| \leq |V_{[i,i^*]}|$, the "**while**" loop takes at most

$$O\left(\sum_{\ell \in [i,\rho(v_i)]} d_G(v_\ell)\right) \text{ time.}$$

Notice that the next iteration of the algorithm is updated to $\rho(v_i)+1$. So in total the algorithm takes at most $O\left(\sum_{v\in V(G)} d_G(v)\right)$ time, i.e., $O(n+m)$ time. Hence we have the following theorem.

Theorem 2. *A minimum* VED*-set of a connected interval graph can be computed in linear-time.*

4. Bipartite Permutation Graphs

In this section, we design a linear-time algorithm for finding a minimum VED-set of a given bipartite permutation graph G. For this algorithm we use dynamic programming paradigm. First, we show an important property of a minimum VED-set of a bipartite permutation graph.

Lemma 6. *Let G be a bipartite permutation graph with a forward-convex ordering $\sigma = (x_1, x_2, \ldots, x_{n_1}, y_1, y_2, \ldots, y_{n_2})$, $\ell = \text{last}(N_G(x_1))$, and $r = \text{last}(N_G(y_1))$. Then there exists a minimum* VED*-set D of G such that either $y_\ell \in D$ or $x_r \in D$.*

Proof. We assume that the graph G is connected and D is a minimum VED-set of G. We have $x_1y_1 \in E(G)$, by Lemma 2(a). If $y_\ell \in D$ or $x_r \in D$, then we are done. Next, we assume that $y_\ell \notin D$ and $x_r \notin D$. Now to ve-dominate the edge x_1y_1, we must have $D \cap (N_G[x_1] \cup N_G[y_1]) \neq \emptyset$. So, we consider four cases.

Case 1. $x_1 \in D$. Note that all the edges incident on any vertex $u \in N_G[x_1]$ are ve-dominated by the vertex x_1 in D. Now let $D' = (D \setminus \{x_1\}) \cup \{x_r\}$. Observe that all the edges incident on the vertices of $X_{[1,r]}$ and $Y_{[1,\text{last}(N_G(x_r))]}$ are ve-dominated by the vertex x_r. Moreover, $N_1(x_1) \subseteq N_1(x_r)$ by Lemma 1(b), that is, $N_G(x_1) \subseteq N_G(x_r)$. It implies that all the edges incident on $N_G[x_1]$ are ve-dominated by the vertex x_r. So D' is a minimum VED-set of G such that $x_r \in D'$.

Case 2. $y_1 \in D$. Note that all the edges incident on the vertices of $N_G[y_1]$ are ve-dominated by the vertex y_1 in D. Let $D' = (D \setminus \{y_1\}) \cup \{y_\ell\}$. Observe that the vertex y_ℓ dominates all the edges incident on the vertices of $Y_{[1,\ell]}$ and $X_{[1,\text{last}(N_G(y_\ell))]}$. Moreover, $N_1(y_1) \subseteq N_1(y_\ell)$ by Lemma 1(a), that is, $N_G(y_1) \subseteq N_G(y_\ell)$. It implies that all the edges incident on $N_G[y_1]$ are ve-dominated by the vertex y_ℓ. So D' is a minimum VED-set of G such that $y_\ell \in D'$.

Case 3. There exists $x_i \in N_G(y_1)$ such that $x_i \in D$. Let $D' = (D \setminus \{x_i\}) \cup \{x_r\}$. Observe that for any vertex $x_i \in N_G(y_1)$, $N_1(x_i) \subseteq N_1(x_r)$ by Lemma 1(b), that is, $N_G(x_i) \subseteq N_G(x_r)$ for $i < r$. It implies that all the edges ve-dominated by x_i are ve-dominated by x_r. So D' is a minimum VED-set of G such that $x_r \in D'$.

Case 4. There exists $y_j \in N(x_1)$ such that $y_j \in D$. Let $D' = (D \setminus \{y_j\}) \cup \{y_\ell\}$. Note that for any vertex $y_j \in N_G(x_1)$, $N_1(y_j) \subseteq N_1(y_\ell)$ by Lemma 1(a), that is, $N_G(y_j) \subseteq N_G(y_\ell)$, for $j < \ell$. It implies that all the edges ve-dominated by y_j are ve-dominated by y_ℓ. So D' is a minimum VED-set of G such that $y_\ell \in D'$.

Thus in each case, we get a minimum VED-set D' such that either $y_\ell \in D'$ or $x_r \in D'$. This completes the proof of the lemma. ∎

Based on the property described in Lemma 6, we have the following lemma which shows a recursive way of computing a minimum VED-set of a bipartite permutation graph.

Lemma 7. *Let G be a bipartite permutation graph with a forward-convex ordering $\sigma = (x_1, x_2, \ldots, x_{n_1}, y_1, y_2, \ldots, y_{n_2})$, $\ell = \text{last}(N_G(x_1))$, $k = \text{last}(N_G(y_\ell))$, $r = \text{last}(N_G(y_1))$, $s = \text{last}(N_G(x_r))$, $\ell' = \text{first}(N_G(x_{k+1}))$, and $r' = \text{first}(N_G(y_{s+1}))$. Then one of the following holds.*

(a) $\gamma_{ve}(G) = \gamma_{ve}(G') + 1$, *where* $G' = G[X_{k+1} \cup Y_{\ell'}]$.

(b) $\gamma_{ve}(G) = \gamma_{ve}(G'') + 1$, *where* $G'' = G[X_{r'} \cup Y_{s+1}]$.

Proof. If D' is a minimum VED-set of G', then it is clear that $D' \cup \{y_\ell\}$ is a VED-set of G. So we have

$$\gamma_{ve}(G) \leq \gamma_{ve}(G') + 1. \tag{1}$$

Similarly if D'' is a minimum VED-set of G'', then it is clear that $D'' \cup \{x_r\}$ is a VED-set of G. So we have

$$\gamma_{ve}(G) \leq \gamma_{ve}(G'') + 1. \tag{2}$$

By Lemma 6, there exists a minimum VED-set D of G such that either $y_\ell \in D$ or $x_r \in D$. So we consider the following two cases.

Case 1. $y_\ell \in D$. Since $y_\ell \in D$, all the edges incident on $N_G[y_\ell]$ are ve-dominated by the vertex y_ℓ. It follows that all the edges incident on vertices $X_{[1,k]}$ are ve-dominated by the vertex y_ℓ. By Lemma 1(a), for all $j < \ell$, $N_1(y_j) \subseteq N_1(y_\ell)$, that is, $N_G(y_j) \subseteq N_G(y_\ell)$. Since $N_G(y_j) \subseteq N_G(y_\ell)$ for any $j < \ell$, all the edges incident on the vertices $Y_{[1,\ell]}$ are ve-dominated by y_ℓ. Moreover, the edges incident on x_k are also ve-dominated by y_ℓ. Let $D' = D \setminus \{y_\ell\}$. By Lemma 2(b) and (d), the set of vertices $\{y_j : j \in [\ell + 1, \ell' - 1]\}$ are isolated vertices and the graph $G' = G[X_{k+1} \cup Y_{\ell'}]$ is a connected bipartite permutation graph. So it is clear that

D' is a VED-set of G'. Thus we have $\gamma_{ve}(G') \leq |D| - 1 = \gamma_{ve}(G) - 1$. Combining with Equation (1), we have $\gamma_{ve}(G) = \gamma_{ve}(G') + 1$.

Case 2. $x_r \in D$. Since $x_r \in D$, all the edges incident on the vertices of $N_G[x_r]$ are ve-dominated by the vertex x_r. It follows that all the edges incident on vertices $Y_{[1,s]}$ are ve-dominated by the vertex x_r. By Lemma 1(b), for all $i < r$, $N_1(x_i) \subseteq N_1(x_r)$, that is, $N_G(x_i) \subseteq N_G(x_r)$. Since $N_G(x_i) \subseteq N_G(x_r)$ for any $i < r$, all the edges incident on the vertices $X_{[1,r]}$ are ve-dominated by x_r. Moreover, the edges incident on y_s are also ve-dominated by x_r. Let $D'' = D \setminus \{x_r\}$. By Lemma 2(c) and (d), the set of vertices $\{x_i \colon i \in [r+1, r'-1]\}$ are isolated vertices and the graph $G'' = G[X_{r'} \cup Y_{s+1}]$ is a connected bipartite permutation graph. So it is clear that D'' is a VED-set of G''. Thus we have $\gamma_{ve}(G'') \leq |D| - 1 = \gamma_{ve}(G) - 1$. Combining with Equation (2), we have $\gamma_{ve}(G) = \gamma_{ve}(G'') + 1$, completing the proof of the lemma. ■

Algorithm 2: VEDS-BPG(G)

Input: A bipartite permutation graph $G = (X, Y, E)$ such that $|X| = n_1$ and $|Y| = n_2$;
Output: A minimum VED-set of G;

```
Initialize D(n1 + 1, n2 + 1) = ∅;
for (i = n1 down to 1) do
    for (j = last(N_G(x_i)) down to first(N_G(x_i))) do
        Let ℓ = last(N_G(x_i)), k = last(N_G(y_ℓ)), r = last(N_G(y_j)) and
          s = last(N_G(x_r)) and ℓ' = first(N_G(x_{k+1})), and
          r' = first(N_G(y_{s+1}));
        Compute D(k + 1, ℓ') and D(r', s + 1);
        if (|D(k + 1, ℓ') ∪ {y_ℓ}| ≤ |D(r', s + 1) ∪ {x_r}|) then
            D(i, j) = D(k + 1, ℓ') ∪ {y_ℓ};
        else
            D(i, j) = D(r', s + 1) ∪ {x_r};
        end
    end
end
return D(1, 1);
```

Based on Lemma 7, we now present a dynamic programming based algorithm to compute a minimum VED-set of a bipartite permutation graph G. For every $i \in [n_1]$ and $j \in [n_2]$, we use the notation $D(i, j)$ to denote the minimum VED-set of $G[X_i \cup Y_j]$. We follow bottom-up approach for computing $D(i, j)$. To apply Lemma 7, we need x_i and y_j to be adjacent. Hence, we process the vertices of X in reverse order of the forward-convex ordering and also for a fixed vertex

in X, we process all its neighbours in reverse order of forward-convex ordering. At each step, we compute $D(i, j)$ according to Lemma 7. At the initial step, with slight abuse of notation, $G[X_{n_1+1} \cup Y_{n_2+1}]$ denote the empty graph and the corresponding $D(n_1 + 1, n_2 + 1) = \emptyset$ for obvious reason. Finally, at the end of Algorithm VEDS-BPG(G), $D(1, 1)$ would return the minimum VED-set of the bipartite permutation graph G. The details are provided below in Algorithm VEDS-BPG(G).

Next, we analyse the running time of Algorithm VEDS-BPG(G). Note that the outer for-loop runs over all vertices of X and for a fixed vertex in X, the inner for-loop runs over all its neighbours. Hence, there are $O(n + m)$ iterations overall. In each iteration, we compute $D(k + 1, \ell')$ and $D(r', s + 1)$ in $O(1)$ time and therefore, we compute $D(i, j)$ in $O(1)$ time. So, Algorithm VEDS-BPG(G) runs in $O(n + m)$ time. Hence, we have the following theorem.

Theorem 3. *A minimum* VED*-set of a bipartite permutation graph can be computed in linear-time.*

5. Conclusion

We proposed two linear-time algorithms for solving the ve-domination problem in interval graphs and bipartite permutation graphs. It would be interesting to investigate this problem in subclasses of chordal graphs such as strongly chordal graphs and directed path graphs. Also, studying the parameterized complexity of the ve-domination problem is another interesting direction of research.

References

[1] K.S. Booth and G.S. Lueker, *Testing for consecutive ones property, interval graphs and graph planarity using PQ-tree algorithms*, J. Comput. System. Sci. **13** (1976) 335–379.
https://doi.org/10.1016/S0022-0000(76)80045-1

[2] R. Boutrig and M. Chellali, *Total vertex-edge domination*, Int. J. Comput. Math. **95** (2018) 1820–1828.
https://doi.org/10.1080/00207160.2017.1343469

[3] R. Boutrig, M. Chellali, T.W. Haynes and S.T. Hedetniemi, *Vertex-edge domination in graphs*, Aequationes Math. **90** (2016) 355–366.
https://doi.org/10.1007/s00010-015-0354-2

[4] X.G. Chen, K. Yin and T. Gao, *A note on independent vertex-edge domination in graphs*, Discrete Optim. **25** (2017) 1–5.
https://doi.org/10.1016/j.disopt.2017.01.002

[5] S. Chitra and R. Sattanathan, *Global vertex-edge domination sets in graphs*, in: Proc. Int. Math. Forum **7** (2012) 233–240.

[6] T.W. Haynes, S.T. Hedetniemi and P.J. Slater, Fundamentals of Domination in Graphs (Marcel Dekker Inc., New York, 1998).

[7] T.W. Haynes, S.T. Hedetniemi and P.J. Slater, Domination in Graphs: Advanced Topics (Marcel Dekker Inc., New York, 1998).

[8] M.A. Henning, S. Pal and D. Pradhan, *Algorithm and hardness results on hop domination in graphs*, Inform. Process. Lett. **153** (2020) 105872. https://doi.org/10.1016/j.ipl.2019.105872

[9] S.K. Jena and G.K. Das, *Vertex-edge domination in unit disk graphs*, in: Proc. of the 6th International Conference on Algorithms and Discrete Applied Mathematics, Lecture Notes in Comput. Sci. **12016** (2020) 67–78. https://doi.org/10.1007/978-3-030-39219-2_6

[10] B. Krishnakumari, M. Chellali and Y.B. Venkatakrishnan, *Double vertex-edge domination*, Discrete Math. Algorithms Appl. **09** (2017) 1750045. https://doi.org/10.1142/S1793830917500458

[11] B. Krishnakumari and Y.B. Venkatakrishnan, *The outer-connected vertex edge domination number of a tree*, Commun. Korean Math. Soc. **33** (2018) 361–369. https://doi.org/10.4134/CKMS.c150243

[12] B. Krishnakumari, Y.B. Venkatakrishnan and M. Krzywkowski, *Bounds on the vertex-edge domination number of a tree*, C. R. Math. Acad. Sci. Paris **352** (2014) 363–366. https://doi.org/10.1016/j.crma.2014.03.017

[13] T.H. Lai and S.S. Wei, *Bipartite permutation graphs with application to the minimum buffer size problem*, Discrete Appl. Math. **74** (1997) 33–55. https://doi.org/10.1016/S0166-218X(96)00014-5

[14] J.R. Lewis, Vertex-Edge and Edge-Vertex Parameters in Graphs, PhD Thesis, Clemson, University, Clemson (2007). https://tigerprints.clemson.edu/all˙dissertations/103

[15] J.R. Lewis, S.T. Hedetniemi, T.W. Haynes and G.H. Fricke, *Vertex-edge domination*, Util. Math. **81** (2010) 193–213.

[16] S. Paul and K. Ranjan, *On vertex-edge and independent vertex-edge domination*, in: Proc. of the 13th International Conference on Combinatorial Optimization and Applications, (Lecture Notes in Comput. Sci. **11949**, 2019) 437–448. https://doi.org/10.1007/978-3-030-36412-0_35

[17] K.W. Peters, Theoretical and Algorithmic Results on Domination and Connectivity, PhD Thesis (Clemson, University Clemson, 1986).

[18] G. Ramalingam and C. Pandu Rangan, *A unified approach to domination problems on interval graphs*, Inform. Process. Lett. **27** (1998) 271–274.
https://doi.org/10.1016/0020-0190(88)90091-9

[19] J. Spinrad, A. Brandstädt and L. Stewart, *Bipartite permutation graphs*, Discrete Appl. Math. **18** (1987) 279–292.
https://doi.org/10.1016/S0166-218X(87)80003-3

[20] R. Ziemann and P. Żyliński, *Vertex-edge domination in cubic graphs*, Discrete Math. **343** (2020) 112075.
https://doi.org/10.1016/j.disc.2020.112075

[21] P. Żyliński, *Vertex-edge domination in graphs*, Aequationes Math. **93** (2019) 735–742.
https://doi.org/10.1007/s00010-018-0609-9

Received 7 October 2020
Revised 5 May 2021
Accepted 15 May 2021
Available online 8 June 2021