

Provided for non-commercial research and educational use. Not for reproduction, distribution or commercial use.
--

Serdica

Mathematical Journal

Сердика

Математическо списание

The attached copy is furnished for non-commercial research and education use only.
Authors are permitted to post this version of the article to their personal websites or institutional repositories and to share with other researchers in the form of electronic reprints.
Other uses, including reproduction and distribution, or selling or licensing copies, or posting to third party websites are prohibited.

For further information on
Serdica Mathematical Journal
which is the new series of
Serdica Bulgaricae Mathematicae Publicationes
visit the website of the journal <http://www.math.bas.bg/~serdica>
or contact: Editorial Office
Serdica Mathematical Journal
Institute of Mathematics and Informatics
Bulgarian Academy of Sciences
Telephone: (+359-2)9792818, FAX:(+359-2)971-36-49
e-mail: serdica@math.bas.bg

AN OPTIMAL ALGORITHM FOR COMPUTING MINIMUM k -HOP DOMINATING SET OF PERMUTATION GRAPHS

Sambhu Charan Barman, Madhumangal Pal, Sukumar Mondal

Communicated by V. Drensky

ABSTRACT. A k -hop dominating set (k -HDS) D of a graph $G = (V, E)$ is a subset of V such that every vertex $x \in V$ is within k -steps from at least one vertex $y \in D$, i.e., $d(x, y) \leq k$, where k is a fixed positive integer. A k -hop dominating set D is said to be minimal if there does not exist any $H \subset D$ such that H is a k -HDS of G . If a dominating set D is minimal as well as it is k -HDS then it is said to be minimum k -hop dominating set. In this paper, we present an optimal algorithm to compute a minimum k -HDS of permutation graphs with n vertices which runs in $O(n)$ time.

1. Introduction. Let $G = (V, E)$ be a simple and undirected graph with n vertices and m edges and let $V = \{1, 2, \dots, n\}$. G is said to be a *permutation graph* iff there exists a permutation $\pi = \{\pi(1), \pi(2), \dots, \pi(n)\}$ on V such that for all $i, j \in V$, $(i, j) \in E$ iff $(i - j)(\pi^{-1}(i) - \pi^{-1}(j)) < 0$, where for each $i \in V$, $\pi^{-1}(i)$ denotes the position of the number i in π [26]. We assume that the graph is connected. Permutation graphs is a sub-class of intersection graphs [26].

2020 *Mathematics Subject Classification.* 05C30, 05C12, 68R10, 68Q25.

Key words: Design & analysis of algorithms, k -hop domination, permutation graphs.

Pnueli et al. [52] showed that this graph is a sub-class of comparability graph. A permutation graph can also be created by its corresponding matching diagram which contains two horizontal parallel lines, named as top line and bottom line. We put the numbers $1, 2, \dots, n$ on the top line, in their order, from left to right, and for each $i = 1, 2, \dots, n$ put the number $\pi(i)$ on the bottom line just below the number i on the top line. Finally, for each i , we draw a straight line joining two i 's on the top line and on the bottom line [26]. We use the same number i to label the resulting line segment. We note that the line i intersects the line j in the diagram iff i and j appear in the reversed order of magnitude in π . That is, lines i and j intersect iff vertices i and j of the corresponding permutation graph are adjacent. Figure 2 shows a permutation graph and Figure 1 is the corresponding matching diagram. We referred these figures throughout our paper.

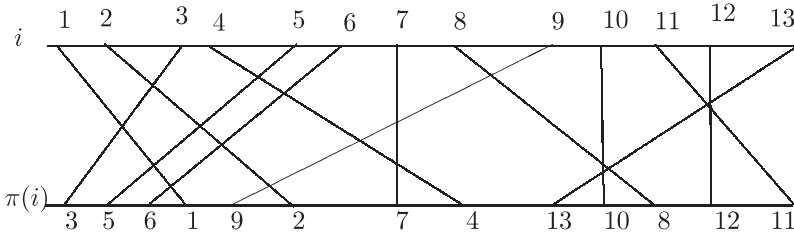


Fig. 1. Matching diagram of the permutation graph G of Figure 2

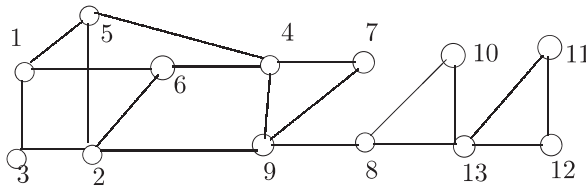


Fig. 2. A permutation graph G

We can set up the matching diagram for a given permutation graph in $O(n^2)$ time only when the graph is in the form of an adjacency list or an adjacency matrix [26, 47]. For this reason, we presume that a permutation representation is given for the input graph. Also, we create two arrays: one is $\pi(j)$, $j = 1, 2, 3, \dots, n$ in which we store the permutation numbers and other array is $\pi^{-1}(j)$, $j = 1, 2, 3, \dots, n$ in which we store the inverse permutation of π . We can compute the array $\pi^{-1}(j)$ from array π in $O(n)$ time.

1.1. Definitions. For a graph $G = (V, E)$, a vertex is said to dominate itself and all of its neighbours. A set $D \subseteq V$ is called a *dominating set* of G if every vertex in V is dominated by at least one vertex in D . Domination and its variations like edge domination [44, 50, 61], weighted domination [13], independent domination [12], connected domination [22], total/open domination [20, 27, 31, 32, 10], locating domination [45], paired-domination [54, 51, 40, 38, 37, 33, 17, 16, 18, 19], k -tuple domination [53, 56, 41, 42] and k -hop domination [57, 39, 24] have been briefly discussed in the literature, see [11, 29, 30, 35].

A *k*-hop dominating set (k -HDS) D of a graph $G = (V, E)$ is a subset of V such that every vertex $x \in V$ is within k -steps from at least one vertex $y \in D$, i.e., $d(x, y) \leq k$, where k is a fixed positive integer. Slater mentions a k -HDS as a k -basis [57] and formulates the k -basis problem in a more general setting called R-dominating set. Ayyaswamy et al. [2] studied about Bounds on the hop domination number of a tree. Also, concept of hop domination was further discussed by Natarajan et al. [48]. A k -hop dominating set D is said to be minimal if there does not exist any $H \subset D$ such that H is a k -HDS of G . If a dominating set D is minimal as well as it is k -HDS then it is said to be minimum k -HDS. For the special case $k = 1$, 1-hop dominating set is called dominating set.

1.2. Applications. Domination plays an important role in graphs theory. It has lot of applications in the real life like radio stations, Modeling of biological networks, facility location problems, land surveying, computer communication networks, problems of locating radar stations, school bus routing, coding theory and kernels of games etc. [1, 6, 7, 8, 9, 34].

Though there are several applications of k -hop domination, an explanation in terms of communication networks will be discussed here. If V represents a set of cities and an edge set E represents communication links, then, as in [43], every body wants to compute a minimum number of cities as sites for transmitting stations so that every city either contains a transmitter or can receive messages from at least one of the transmitting stations through the links. If only direct transmissions are acceptable, then one wishes to obtain a minimum 1-hop dominating set. If communication over paths of k links (but not of $k + 1$ links) is adequate in quality and rapidity, the problem becomes that of determining a k -HDS with fewest possible vertices.

1.3. Survey of the related works. In graph theory, many researchers broadly studied domination problems to solve many real life problems. Few years ago, a book [30] was written on the topic of domination to records more than 1,200 research papers on domination in graphs. After the publication of this book, lot

of papers on the topic of domination have been published. Historically, the first domination-type problem came from chess. In 1862, the chess master C.F. de Jaenisch wrote a paper [23] on the applications of mathematical analysis to chess. Chang [11] has deeply researched about domination in graphs. Furthermore various types of dominations are discussed in [21, 27, 28]. An optimal algorithm for finding the minimum cardinality dominating set on permutation graphs was presented in [14]. Another type of dominating set has been widely studied in [20, 27, 31, 32] is a total dominating set. Sumner and Blitch [58] studied the domination number of graphs and showed that if any edge is added to a graph then its domination number is changed. Also, Haynes, Mynhardt and van der Merwe [29, 60, 31, 32] studied the total domination edge critical graphs.

Among the variations of domination, the k -hop domination has been briefly discussed in the literature, see [11, 27, 29, 30, 33, 35, 44, 61, 54, 2, 48]. Demaine et al. [24] considered the class of planar (and some others) graph G and gave an algorithm to compute an optimal k -hop dominating set in $O(n^4)$ -time, where n is the number of vertices of G . Basuchowdhuri et al. [5] applied k -HDS to compute influential nodes in social network and proved that the decision version of the k -HDS problem is NP-complete. After that, Henning et al. [36] showed that the k -HDS problem is NP-complete for planar bipartite graphs and planar chordal graphs. Also, Farhadi et al. [25] proved that the k -HDS problem is NP-complete for planar bipartite graphs and planar chordal graphs. Natara-jan et al. [48] discussed about the equality of hop domination number and other domination parameters like connected domination number and total domination number on different graph classes. Furthermore, Ayyaswamy et al. [2] discussed about Bounds on the hop domination number of a tree. Also, Rana et al. [55] have designed an efficient algorithm to solve the distance k -domination problem on permutation graphs. Kundu et al. [39] have designed an $O(n)$ time algorithm for finding an optimal k -hop dominating set of a tree. This is an extensive work of the previous result for an optimal 1-hop dominating set for a tree [20]. Recently, Barman et al. [4] have designed an $O(n)$ time algorithm to find minimum k -hop dominating set of interval graphs.

1.4. Main result. In this paper, we present an optimal algorithm to find a minimum k -HDS of the undirected and connected permutation graph in $O(n)$ time, where n is the number of vertices of the permutation graph. This problem is similar to finding a dominating set, but differs in terms of number of hops needed to reach all the nodes. We have generalized the problem as k -hop dominating set problem on permutation graphs.

1.5. Organization of the paper. In the next section, we discuss about the construction of BFS tree $T^*(1)$ with root as 1 of permutation graphs. Some notations related to our problem are also presented in this section. In Section 3, we prove some important results related to the *k*-hop dominating set of permutation graphs. An algorithm is presented to find a minimum *k*-hop dominating set of permutation graphs in Section 4. The time complexity is also calculated in this section.

In the next section, we discuss about the construction of BFS tree $T^*(1)$ with root as 1 of permutation graphs.

2. BFS tree and main path on it. It is well known that the BFS is an important graph traversal technique. Also, it constructs a BFS tree. In BFS, started with vertex *v* as root, we first visit all the neighbors of the root *v*. Then for each of those neighbors of *v*, we visit their unvisited neighbor nodes. We keep going until all vertices reachable from the root *v* have been visited.

The BFS tree can be constructed on general graphs in $O(n + m)$ time, where *n* and *m* represent respectively the number of vertices and number of edges [59]. Chen et al. [15] gave a BFS algorithm on tree. Furthermore, Olariu [49]

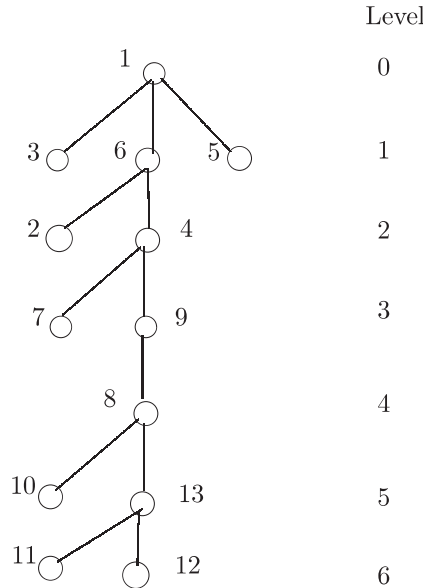


Fig. 3. BFS tree $T^*(1)$ of permutation graph *G* of Figure 2

constructed a BFS tree with root as n (the highest numbered vertex), which is known as *interval tree*. Mondal et al. [46] have designed an algorithm to construct a BFS tree $T^*(i)$ with root as i on trapezoid graph in $O(n)$ time, where n is the number of vertices. Also, Barman et al. [3] have designed an $O(n)$ time **Algorithm PBFS** to construct a BFS tree $T^*(x)$ with root as x on permutation graph $G = (V, E)$, where n is the number of vertices. The BFS tree $T^*(1)$ rooted at 1 of the permutation graph of Figure 2 is shown in Figure 3.

We define the level of a vertex v as a distance of v from the root 1 of the tree $T^*(1)$ and denoted by $level(v)$, $v \in V$ and take the level of root 1 as 0. The level of each vertex on BFS tree $T^*(1)$, $1 \in V$ can be assigned by the BFS algorithm of Chen and Das [15].

2.1. Computation of main path on BFS tree $T^*(1)$. In BFS tree $T^*(1)$ with root as 1, let z be any vertex at level h , where h is the height of tree $T^*(1)$. Also we assume that the shortest path between 1 and z which is $z \rightarrow parent(z) \rightarrow parent(parent(z)) \rightarrow \dots \rightarrow 1$ as the *main path* between 1 and z .

Before going to the algorithm, we present some notations which are needed through out the paper.

2.2. Some notations. Here we present some notations which are needed to designed the algorithm.

- u_i^* : u_i^* is the vertex on the main path of the BFS tree $T^*(1)$ at level i .
- $N(u)$: $N(u)$ is the open neighbourhood set of the vertex u ,
i.e., $N(u) = \{x : (x, u) \in E\}$.
- $N[u]$: $N[u]$ is the closed neighbourhood set of the vertex u ,
i.e., $N[u] = N(u) \cup \{u\}$.
- N_i : N_i is the set of members at level i on the BFS tree $T^*(1)$.
- X_i : $X_i = N_i - \{u_i^*\}$.
- h : the height of the BFS tree $T^*(1)$.
- $d(u, v)$: the shortest distance between two vertices u and v .

Before going to our proposed algorithm we prove the following important results relating to minimum k -HDS D of permutation graphs.

3. Some results relating to minimum k -HDS of permutation graphs. In this section we present some important results relating to minimum k -HDS D of permutation graphs.

Lemma 3.1. *If there exists at least one vertex of X_1 which is not connected with u_1^* and u_2^* , then $d(u_{k-1}^*, x) \leq k$ for all $x \in \cup_{i=0}^{k-1} N_i$.*

Proof. Let x_i be any member of X_i for $i = 1, 2, \dots, h$ and v be a member of X_1 such that $(v, u_1^*) \notin E$ and $(v, u_2^*) \notin E$. Since $\text{parent}(v) = u_0^*$, so, $d(u_{k-1}^*, v) = k$ (as $v \rightarrow u_0^* \rightarrow u_1^* \rightarrow u_2^* \rightarrow \dots \rightarrow u_{k-1}^*$). Also, $d(u_{k-1}^*, u_0^*) = k - 1 \leq k$ (as $u_0^* \rightarrow u_1^* \rightarrow u_2^* \rightarrow \dots \rightarrow u_{k-1}^*$) and $d(u_{k-1}^*, x_1) \leq k$ (as $x_1 \rightarrow u_0^* \rightarrow u_1^* \rightarrow u_2^* \rightarrow \dots \rightarrow u_{k-1}^*$ or $x_1 \rightarrow u_1^* \rightarrow u_2^* \rightarrow \dots \rightarrow u_{k-1}^*$ or $x_1 \rightarrow u_2^* \rightarrow u_3^* \rightarrow \dots \rightarrow u_{k-1}^*$) for all $x_1 (\neq v) \in X_1$. Now, $d(u_{k-1}^*, x_i) \leq k + 1 - i$ (as $x_i \rightarrow u_{i-1}^* \rightarrow u_i^* \rightarrow u_{i+1}^* \rightarrow \dots \rightarrow u_{k-1}^*$ or $x_i \rightarrow u_i^* \rightarrow u_{i+1}^* \rightarrow \dots \rightarrow u_{k-1}^*$ or $x_i \rightarrow u_{i+1}^* \rightarrow u_{i+2}^* \rightarrow \dots \rightarrow u_{k-1}^*$), for $i = 2, 3, \dots, (k - 2)$. Furthermore, $d(u_{k-1}^*, x_{k-1}) \leq 2$ (as $x_{k-1} \rightarrow u_{k-2}^* \rightarrow u_{k-1}^*$ or $x_{k-1} \rightarrow u_{k-1}^*$). Therefore, $d(u_{k-1}^*, x) \leq k$ for all $x \in \cup_{i=0}^{k-1} N_i$. $\square$

Lemma 3.2. *If $X_1 = \phi$ or each member of X_1 is adjacent with u_1^* or u_2^* , then $d(u_k^*, x) \leq k$ for all $x \in \cup_{i=0}^{2k} N_i$.*

Proof. Let x_i be any member of X_i for $i = 0, 1, 2, \dots, h$. So, $d(u_k^*, u_0^*) = k$ (as $u_0^* \rightarrow u_1^* \rightarrow u_2^* \rightarrow \dots \rightarrow u_k^*$). Now, if $X_1 = \phi$, then $d(u_k^*, x_i) \leq k + 2 - i$ (as $x_i \rightarrow u_{i-1}^* \rightarrow u_i^* \rightarrow u_{i+1}^* \rightarrow \dots \rightarrow u_k^*$ or $x_i \rightarrow u_i^* \rightarrow u_{i+1}^* \rightarrow u_{i+2}^* \rightarrow \dots \rightarrow u_k^*$ or $x_i \rightarrow u_{i+1}^* \rightarrow u_{i+2}^* \rightarrow \dots \rightarrow u_k^*$), for $i = 2, 3, \dots, (k - 1)$. Furthermore, $d(u_k^*, x_k) \leq 2$ (as $x_k \rightarrow u_{k-1}^* \rightarrow u_k^*$ or $x_k \rightarrow u_k^*$). Therefore, $d(u_k^*, x) \leq k$ for all $x \in \cup_{i=0}^k N_i$.

Again, if each member of X_1 is adjacent with u_1^* or u_2^* , then $d(u_k^*, x_1) \leq k$ (as $x_1 \rightarrow u_1^* \rightarrow u_2^* \rightarrow u_3^* \rightarrow \dots \rightarrow u_k^*$ or $x_1 \rightarrow u_2^* \rightarrow u_3^* \rightarrow \dots \rightarrow u_k^*$). Also, $d(u_k^*, x_i) \leq k + 2 - i$ (as $x_i \rightarrow u_{i-1}^* \rightarrow u_i^* \rightarrow u_{i+1}^* \rightarrow \dots \rightarrow u_k^*$ or $x_i \rightarrow u_i^* \rightarrow u_{i+1}^* \rightarrow u_{i+2}^* \rightarrow \dots \rightarrow u_k^*$ or $x_i \rightarrow u_{i+1}^* \rightarrow u_{i+2}^* \rightarrow \dots \rightarrow u_k^*$), for $i = 2, 3, \dots, (k - 1)$. Furthermore, $d(u_k^*, x_k) \leq 2$ (as $x_k \rightarrow u_{k-1}^* \rightarrow u_k^*$ or $x_k \rightarrow u_k^*$). Therefore, $d(u_k^*, x) \leq k$ for all $x \in \cup_{i=0}^k N_i$.

Again, from the **Algorithm PBFS**, we know that $\text{parent}(x_i) = u_{i-1}^*$, for all $i = 1, 2, \dots, h$. So, $d(u_k^*, x_{k+i}) = i$ (as $u_k^* \rightarrow u_{k+1}^* \rightarrow u_{k+2}^* \rightarrow \dots \rightarrow u_{k+i-1}^* \rightarrow x_{k+i}$), for $i = 1, 2, 3, \dots, k$. Also, $d(u_k^*, u_{2k}^*) = k$ (as $u_k^* \rightarrow u_{k+1}^* \rightarrow u_{k+2}^* \rightarrow \dots \rightarrow u_{2k}^*$). Hence, $d(u_k^*, x) \leq k$ for all $x \in \cup_{i=k+1}^{2k} N_i$. Therefore, $d(u_k^*, x) \leq k$ for all $x \in \cup_{i=0}^{2k} N_i$. $\square$

Lemma 3.3. *If at any stage, u_l^* is selected as a member of D , then $d(x, u_l^*) \leq k$ for all $x \in \cup_{i=1}^k N_{l+i}$.*

Proof. Let at any stage, u_l^* be the last selected member of D . Also, we assume that x_i is any member of N_i for $i = 1, 2, \dots, h$. Again, from **Algorithm PBFS**, we know that $\text{parent}(x_i) = u_{l+i-1}^*$ for $i = 1, 2, 3, \dots, h$. So, $d(u_l^*, x_{l+i}) = i$ (as $u_l^* \rightarrow u_{l+1}^* \rightarrow u_{l+2}^* \rightarrow \dots \rightarrow u_{l+i-1}^* \rightarrow x_{l+i}$), for $i = 1, 2, 3, \dots, k$. Hence,

$d(x, u_l^*) \leq k$ for all $x \in \cup_{i=1}^k N_{l+i}$. $\square$

Lemma 3.4. *If at any stage, u_l^* is selected as a member of D and if $X_{l+k+1} = \phi$ or $(x, u_{l+k+1}^*) \in E$ or $(x, u_{l+k+2}^*) \in E$ for all $x \in X_{l+k+1}$ and u_{l+2k}^* exists, then u_{l+2k}^* is a possible member of D .*

Proof. Let at any stage, u_l^* be the last selected member of D . Also, let x_i be any member of X_i for $i = 1, 2, \dots, h$. So, by Lemma 3.3, we have $d(x, u_l^*) \leq k$ for all $x \in \cup_{i=1}^k N_{l+i}$. Furthermore, $d(u_{l+2k}^*, u_{l+k+1}^*) = k - 1$ (as $u_{l+k+1}^* \rightarrow u_{l+k+2}^* \rightarrow u_{l+k+3}^* \rightarrow \dots \rightarrow u_{l+2k}^*$). Now, if $X_{l+k+1} = \phi$, then $d(u_{l+2k}^*, x_{l+k+i}) \leq (k+2-i)$ (as $u_{l+2k}^* \rightarrow u_{l+2k-1}^* \rightarrow u_{l+2k-2}^* \rightarrow \dots \rightarrow u_{l+k+i-1}^* \rightarrow x_{l+k+i}$ or $u_{l+2k}^* \rightarrow u_{l+2k-1}^* \rightarrow u_{l+2k-2}^* \rightarrow \dots \rightarrow u_{l+k+i}^* \rightarrow x_{l+k+i}$ or $u_{l+2k}^* \rightarrow u_{l+2k-1}^* \rightarrow u_{l+2k-2}^* \rightarrow \dots \rightarrow u_{l+k+i+1}^* \rightarrow x_{l+k+i}$) for $i = 2, 3, 4, \dots, (k-1)$. Again, $d(u_{l+2k}^*, x_{l+2k}) \leq 2$ (as $u_{l+2k}^* \rightarrow u_{l+2k-1}^* \rightarrow x_{l+2k}$ or $u_{l+2k}^* \rightarrow x_{l+2k}$). So, if $X_{l+k+1} = \phi$, then u_{l+2k}^* is a possible member of D .

Again, if $(x, u_{l+k+1}^*) \in E$ or $(x, u_{l+k+2}^*) \in E$ for all $x \in X_{l+k+1}$, then $d(u_{l+2k}^*, x) \leq k$ (as $u_{l+2k}^* \rightarrow u_{l+2k-1}^* \rightarrow u_{l+2k-2}^* \rightarrow \dots \rightarrow u_{l+k+1}^* \rightarrow x$ or $u_{l+2k}^* \rightarrow u_{l+2k-1}^* \rightarrow u_{l+2k-2}^* \rightarrow \dots \rightarrow u_{l+k+2}^* \rightarrow x$). Also, $d(u_{l+2k}^*, x_{l+k+i}) \leq (k+2-i)$ (as $u_{l+2k}^* \rightarrow u_{l+2k-1}^* \rightarrow u_{l+2k-2}^* \rightarrow \dots \rightarrow u_{l+k+i-1}^* \rightarrow x_{l+k+i}$ or $u_{l+2k}^* \rightarrow u_{l+2k-1}^* \rightarrow u_{l+2k-2}^* \rightarrow \dots \rightarrow u_{l+k+i}^* \rightarrow x_{l+k+i}$ or $u_{l+2k}^* \rightarrow u_{l+2k-1}^* \rightarrow u_{l+2k-2}^* \rightarrow \dots \rightarrow u_{l+k+i+1}^* \rightarrow x_{l+k+i}$) for $i = 2, 3, 4, \dots, (k-1)$. Again, $d(u_{l+2k}^*, x_{l+2k}) \leq 2$ (as $u_{l+2k}^* \rightarrow u_{l+2k-1}^* \rightarrow x_{l+2k}$ or $u_{l+2k}^* \rightarrow x_{l+2k}$). So, if $(x, u_{l+k+1}^*) \in E$ for all $x \in X_{l+k+1}$, then u_{l+2k}^* is a possible member of D . Therefore, if $X_{l+k+1} = \phi$ or $(x, u_{l+k+1}^*) \in E$ or $(x, u_{l+k+2}^*) \in E$ for all $x \in X_{l+k+1}$ and u_{l+2k}^* exists, then u_{l+2k}^* is a possible member of D . $\square$

Lemma 3.5. *If at any stage, u_l^* is selected as a member of D and if $(x, u_{l+k+2}^*) \in E$ for all $x \in (X_{l+k+1} \cup X_{l+k+2})$ or if $(x, u_{l+k+2}^*) \in E$ for all $x \in X_{l+k+1}$ and $(y, u_{l+k+3}^*) \in E$ for all $y \in X_{l+k+2}$ or if $X_{l+k+1} \cup X_{l+k+2} = \phi$ and u_{l+2k+1}^* exists, then u_{l+2k+1}^* is a possible member of D .*

Proof. Let at any stage, u_l^* be the last selected member of D . Also, let x_i be any member of X_i for $i = 1, 2, \dots, h$. So, by Lemma 3.3, we have $d(x, u_l^*) \leq k$ for all $x \in \cup_{i=1}^k N_{l+i}$. Furthermore, $d(u_{l+2k+1}^*, u_{l+k+1}^*) = k$ (as $u_{l+k+1}^* \rightarrow u_{l+k+2}^* \rightarrow u_{l+k+3}^* \rightarrow \dots \rightarrow u_{l+2k+1}^*$). Now, if $X_{l+k+1} \cup X_{l+k+2} = \phi$, then $d(u_{l+2k+1}^*, x_{l+k+i}) \leq (k+3-i)$ (as $u_{l+2k+1}^* \rightarrow u_{l+2k}^* \rightarrow u_{l+2k-1}^* \rightarrow \dots \rightarrow u_{l+k+i-1}^* \rightarrow x_{l+k+i}$ or $u_{l+2k+1}^* \rightarrow u_{l+2k}^* \rightarrow u_{l+2k-1}^* \rightarrow \dots \rightarrow u_{l+k+i}^* \rightarrow x_{l+k+i}$ or $u_{l+2k+1}^* \rightarrow u_{l+2k}^* \rightarrow u_{l+2k-1}^* \rightarrow \dots \rightarrow u_{l+k+i+1}^* \rightarrow x_{l+k+i}$) for $i = 3, 4, \dots, k$. Also, $d(u_{l+2k+1}^*, x_{l+2k+1}) \leq 2$ (as $u_{l+2k+1}^* \rightarrow u_{l+2k}^* \rightarrow x_{l+2k+1}$ or

$u_{l+2k+1}^* \rightarrow x_{l+2k+1}$). So, if $X_{l+k+1} \cup X_{l+k+2} = \phi$, then u_{l+2k+1}^* is a possible member of D . Again, if $(x, u_{l+k+2}^*) \in E$ for all $x \in (X_{l+k+1} \cup X_{l+k+2})$, then $d(u_{l+2k+1}^*, x) = k$ (as $u_{l+2k+1}^* \rightarrow u_{l+2k}^* \rightarrow u_{l+2k-1}^* \rightarrow \dots \rightarrow u_{l+k+2}^* \rightarrow x$). Also, $d(u_{l+2k+1}^*, x_{l+k+i}) \leq (k+3-i)$ (as $u_{l+2k+1}^* \rightarrow u_{l+2k}^* \rightarrow u_{l+2k-1}^* \rightarrow \dots \rightarrow u_{l+k+i-1}^* \rightarrow x_{l+k+i}$ or $u_{l+2k+1}^* \rightarrow u_{l+2k}^* \rightarrow u_{l+2k-1}^* \rightarrow \dots \rightarrow u_{l+k+i}^* \rightarrow x_{l+k+i}$ or $u_{l+2k+1}^* \rightarrow u_{l+2k}^* \rightarrow u_{l+2k-1}^* \rightarrow \dots \rightarrow u_{l+k+i+1}^* \rightarrow x_{l+k+i}$) for $i = 3, 4, \dots, k$. Furthermore, $d(u_{l+2k+1}^*, x_{l+2k+1}) \leq 2$ (as $u_{l+2k+1}^* \rightarrow u_{l+2k}^* \rightarrow x_{l+2k+1}$ or $u_{l+2k+1}^* \rightarrow x_{l+2k+1}$). So, if $(x, u_{l+k+2}^*) \in E$ for all $x \in (X_{l+k+1} \cup X_{l+k+2})$, then u_{l+2k+1}^* is a possible member of D . Hence, u_{l+2k+1}^* is a possible member of D . Again, if $(x, u_{l+k+2}^*) \in E$ for all $x \in X_{l+k+1}$ and $(y, u_{l+k+3}^*) \in E$ for all $y \in X_{l+k+2}$, then $d(u_{l+2k+1}^*, x) = k$ (as $u_{l+2k+1}^* \rightarrow u_{l+2k}^* \rightarrow u_{l+2k-1}^* \rightarrow \dots \rightarrow u_{l+k+2}^* \rightarrow x$) and $d(u_{l+2k+1}^*, y) = k-1$ (as $u_{l+2k+1}^* \rightarrow u_{l+2k}^* \rightarrow u_{l+2k-1}^* \rightarrow \dots \rightarrow u_{l+k+3}^* \rightarrow y$). Also, $d(u_{l+2k+1}^*, x_{l+k+i}) \leq (k+3-i)$ (as $u_{l+2k+1}^* \rightarrow u_{l+2k}^* \rightarrow u_{l+2k-1}^* \rightarrow \dots \rightarrow u_{l+k+i-1}^* \rightarrow x_{l+k+i}$ or $u_{l+2k+1}^* \rightarrow u_{l+2k}^* \rightarrow u_{l+2k-1}^* \rightarrow \dots \rightarrow u_{l+k+i}^* \rightarrow x_{l+k+i}$ or $u_{l+2k+1}^* \rightarrow u_{l+2k}^* \rightarrow u_{l+2k-1}^* \rightarrow \dots \rightarrow u_{l+k+i+1}^* \rightarrow x_{l+k+i}$) for $i = 3, 4, \dots, k$. Furthermore, $d(u_{l+2k+1}^*, x_{l+2k+1}) \leq 2$ (as $u_{l+2k+1}^* \rightarrow u_{l+2k}^* \rightarrow x_{l+2k+1}$ or $u_{l+2k+1}^* \rightarrow x_{l+2k+1}$). So, if $(x, u_{l+k+2}^*) \in E$ for all $x \in X_{l+k+1}$ and $(y, u_{l+k+3}^*) \in E$ for all $y \in X_{l+k+2}$, then u_{l+2k+1}^* is a possible member of D . Hence, u_{l+2k+1}^* is a possible member of D . $\square$

Lemma 3.6. *If at any stage, u_l^* is selected as a member of D and u_{l+2k-1}^* exists, then u_{l+2k-1}^* is a possible member of D .*

Proof. Let at any stage, u_l^* be the last selected member of D . Also, let x_i be any member of X_i for $i = 1, 2, \dots, h$. Now, by Lemma 3.3, we have $d(x, u_l^*) \leq k$ for all $x \in \cup_{i=1}^k N_{l+i}$. Again, $d(u_{l+2k-1}^*, u_{l+k+1}^*) = k-2$ (as $u_{l+k+1}^* \rightarrow u_{l+k+2}^* \rightarrow u_{l+k+3}^* \rightarrow \dots \rightarrow u_{l+2k-1}^*$). Also, $d(u_{l+2k-1}^*, x_{l+k+i}) \leq (k+1-i)$ (as $u_{l+2k-1}^* \rightarrow u_{l+2k-2}^* \rightarrow u_{l+2k-3}^* \rightarrow \dots \rightarrow u_{l+k+i-1}^* \rightarrow x_{l+k+i}$ or $u_{l+2k-1}^* \rightarrow u_{l+2k-2}^* \rightarrow u_{l+2k-3}^* \rightarrow \dots \rightarrow u_{l+k+i}^* \rightarrow x_{l+k+i}$ or $u_{l+2k-1}^* \rightarrow u_{l+2k-2}^* \rightarrow u_{l+2k-3}^* \rightarrow \dots \rightarrow u_{l+k+i+1}^* \rightarrow x_{l+k+i}$) for $i = 1, 2, 3, 4, \dots, (k-2)$. Furthermore, $d(u_{l+2k-1}^*, x_{l+2k-1}) \leq 2$ (as $u_{l+2k-1}^* \rightarrow u_{l+2k-2}^* \rightarrow x_{l+2k-1}$ or $u_{l+2k-1}^* \rightarrow x_{l+2k-1}$). Therefore, if u_{l+2k-1}^* exists, then u_{l+2k-1}^* is a possible member of D . $\square$

Lemma 3.7. *If at any stage, u_l^* is selected as a member of D and if $k < |h-l| \leq 2k-1$, then u_{l+i}^* is a possible member of D , where $i = k, k+1, k+2, \dots, 2k-1$.*

Proof. Let at any stage, u_l^* be the last selected member of D and $k < |h-l| \leq 2k-1$. Now, $d(x, u_l^*) \leq k$ for all $x \in \cup_{i=1}^k N_{l+i}$ (by Lemma

3.3). Also, By Lemma 3.6, we know that if u_{l+2k-1}^* exists, then u_{l+2k-1}^* is a possible member of D . Obviously, u_{l+i}^* is a possible member of D , where $i = k, k+1, k+2, \dots, 2k-1$. $\square$

4. Algorithm and its Complexity. From the above lemmas, it is observed that if u_l^* is selected as a member of D at any stage, then either u_{l+2k+1}^* or u_{l+2k}^* or u_{l+2k-1}^* will be selected as a new member of D at next stage. Also, it is observed that the vertex u_{l+2k-1}^* can be considered as a member of D at any circumstance after selection of u_l^* . But, our aim is to compute the set D such that $|D|$ is minimum. To find D with minimum cardinality we select u_{l+2k+1}^* (as first preference) or u_{l+2k}^* , if possible. All possible cases for selection of the members of D are already presented in terms of lemmas. Now, a procedure FINDNEXT is formally presented in the following which computes the level L of next vertex u_L^* of D , if the level l of the currently selected vertex u_l^* is supplied.

Procedure FINDNEXT(l, L)

//This procedure computes the level L such that u_L^* is the next member of D where as u_l^* is the currently selected vertex of D . The sets X_i and the array $u_i^*, i = 0, 1, 2, \dots, h$, are known globally. In this procedure $h > l + k$.//

Initially $L = l$;

if $(x, u_{l+k+2}^*) \in E$ for all $x \in (X_{l+k+1} \cup X_{l+k+2})$ or if $(x, u_{l+k+2}^*) \in E$ for all $x \in X_{l+k+1}$ and $(y, u_{l+k+3}^*) \in E$ for all $y \in X_{l+k+2}$ or if $X_{l+k+1} \cup X_{l+k+2} = \phi$ and u_{l+2k+1}^* exists, then $L = l + 2k + 1$; (Lemma 3.5)

else if $X_{l+k+2} = \phi$ or $(x, u_{l+k+1}^*) \in E$ or $(x, u_{l+k+2}^*) \in E$ for all $x \in X_{l+k+1}$ and u_{l+2k}^* exists, then $L = l + 2k$; (Lemma 3.4)

else if u_{l+2k-1}^* exists, then $L = l + 2k - 1$; (Lemma 3.6)

else $L = l + k + i$, for any $i = 0, 1, 2, \dots, k - 2$; (Lemma 3.7)

endif;

return L ;

end FINDNEXT

Now, we are going to present the complete algorithm to compute a minimum k -HDS D of permutation graphs. Using repeatedly, the procedure FINDNEXT, one can compute minimum k -HDS of permutation graphs. The complete algorithm is presented below.

Algorithm MKHDSP

Input: A permutation representation $i, \pi(i)$ of a permutation graph G , for $i = 1, 2, 3, \dots, n$.

Output: Minimum k -hop dominating set D of the permutation graph G .

Initially $D = \phi$ (empty set) and $u_0^* = 1$, the root of the BFS tree $T^*(1)$.

Step 1: Construct the BFS tree $T^*(1)$.

Step 2: Compute the vertices on the main path of the BFS tree $T^*(1)$ and let them be $u_i^*, i = 0, 1, 2, \dots, h$.

Step 3: Compute the sets $N_i, i = 0, 1, 2, \dots, h$.

Step 4: If $h \leq k$ then $D = D \cup \{u_l^*\}$, where l is any one of $\{0, 1, 2, \dots, h\}$.
(Using Lemma 3.1)

else if $X_1 = \phi$ or each member of X_1 is adjacent with u_1^* or u_2^* then $l = k$

else $l = k - 1$;

endif;

$D = D \cup \{u_l^*\}$

Step 5: Repeat

Call FINDNEXT(l, L); //Find level L for the next member of D //

$l = L$;

$D = D \cup \{u_l^*\}$;

Until $(|h - l| \leq k)$;

end MKHDSP.

Using **Algorithm MKHDSP**, we get a minimum 2-hop dominating set $D = \{6, 12\}$ of the permutation graph G of Figure 2.

4.1. Illustrative Example. To illustrate the execution process of **Algorithm MKHDSP**, we consider a permutation graph G of Figure 2 with its permutation representation (i.e., matching diagram) of Figure 1 for finding a minimum 2-HDS D of the graph G . Let us assume that $k = 2$ and $D = \phi$. In Step 1, we construct a BFS tree $T^*(1)$ of Figure 3, using the **Algorithm PBFS** [3]. In the next Step, we find the members of the main path which are $u_0^* = 1, u_1^* = 6, u_2^* = 4, u_3^* = 9, u_4^* = 8, u_5^* = 13$, and $u_6^* = 12$, where $h = 6$. In Step 3, we compute the sets $N_0 = \{1\}$, $N_1 = \{3, 6, 5\}$, $N_2 = \{2, 4\}$, $N_3 = \{7, 9\}$, $N_4 = \{8\}$, $N_5 = \{10, 13\}$ and $N_6 = \{11, 12\}$. In Step 4, we see that $h = 6 \geq k = 2$ and $X_1 = \{3, 5\} \neq \phi$. Also, $(3, 6) \notin E$, $(5, 6) \notin E$ and $(3, 4) \notin E$, where $u_2^* = 4$. So, $l = k - 1 = 2 - 1 = 1$ and $D = \{u_l^*\}$, i.e., $D = \{6\}$. So, all the members of $\cup_{i=0}^3 N_i$ are within 2 steps from the vertex 6. In Step 5, we call the **Procedure** FINDNEXT (l, L), to find the next member of D at each iteration. At first iteration of the **Repeat** loop, we set $L = l + 2k + 1 = 6$ as $X_{l+k+1} = X_4 = \phi$

and $(10, 13) \in E$. After that, we set $l = L = 6$. So, $D = D \cup \{u_l^*\} = \{2, 6\}$. Now, $h - l = 6 - 6 = 0 < k$, so, **Repeat** loop is stopped. Hence, minimum 2-hop dominating set of the graph G is $D = \{6, 12\}$.

Lemma 4.1. *The set D is a minimum k -hop dominating set.*

Proof. In **Algorithm MKHDSP**, we select the first member u_l^* of D in such a way that either every vertex $x \in V$ is within k -steps from u_l^* or maximum number of vertices $(\cup_{i=0}^{2k} N_i)$ are within k -steps from u_l^* . After the selection of u_l^* , we call the **Procedure FINDNEXT**(l, L) to find out the next member of D . In each calling, we have three choices $(u_{l+2k+1}^*, u_{l+2k}^*, u_{l+2k-1}^*)$ to select the next member of D . But, we always select u_{l+2k+1}^* (if possible) as the next member of D . If it is not possible, then we select u_{l+2k}^* (if possible) as the next member of D . Otherwise, we select u_{l+2k-1}^* as the next member of D . Hence, D is a minimum k -hop dominating set. $\square$

Theorem 4.2. *The time complexity to find a minimum k -hop dominating set D of a connected permutation graph is $O(n)$, where n is the number of vertices of the permutation graph.*

Proof. In Step 1, the BFS tree $T^*(1)$ can be constructed in $O(n)$ time [3]. Since the main path contains only $h + 1$ vertices, so the main path (in Step 2) can be marked in $O(n)$ time. Step 3, i.e., the sets $N_i, i = 0, 1, 2, \dots, h$ can be computed in $O(n)$ time as $N_i, i = 0, 1, 2, \dots, h$ are mutually exclusive. Time complexity is needed to perform Step 4 is $O(n)$. In Step 5, we call the **Procedure FINDNEXT**(l, L). Each iteration of **Repeat** loop takes maximum $O(|\cup_{i=0}^{2k+1} N_{l+i}|)$ -time for a given l . The **Algorithm MKHDSP** calls the procedure **FINDNEXT** for $|D| - 1$ times and each time the value of l is increased by either $2k - 1$ or $2k$ or $2k + 1$. Also, if the vertices of the set $\cup_{i=0}^{2k+1} N_{l+i}$ (or $\cup_{i=0}^{2k+1} N_{L+i}$) are considered to find the j th (or $(j + 1)$ th) member of D , then $\cup_{i=0}^{2k+1} N_{l+i}$ and $\cup_{i=0}^{2k+1} N_{L+i}$ are disjoint. Therefore, Step 5 takes $O(|\cup_{i=l}^h N_i|) = O(n)$ time. Hence, overall time complexity is $O(n)$. $\square$

5. Conclusion. Domination in graphs has received considerable attention in graph theory. Among the variations of dominations, k -hop domination has been briefly discussed in the past and it has important applications in communication networks. In this paper, we proposed an $O(n)$ time algorithm to find a minimum k -hop dominating set of simple, undirected and connected permutation graphs. We feel that it would be interesting to design an $O(n)$ time algorithm to

find a minimum k -hop dominating set of weighted permutation graphs, trapezoid graphs and interval graphs.

Acknowledgment. We thank the editors and anonymous reviewers for their valuable comments and suggestions to improve our paper.

REFERENCES

- [1] B. AWERBUCH, D. PELEG. Sparse partitions (extended abstract). In: 31st Annual Symposium on Foundations of Computer Science, 1990, 503–513, <https://doi.org/10.1109/FSCS.1990.89571>.
- [2] S. K. AYYASWAMY, B. KRISHNAKUMARI, C. NATARAJAN, Y. B. VENKATAKRISHNAN. Bounds on the hop domination number of a tree. *Proc. Indian Acad. Sci. Math. Sci.* **125**, 4 (2015), 449–455.
- [3] S. C. BARMAN, M. PAL, S. MONDAL. An efficient algorithm to find next-to-shortest path on permutation graphs. *J. Appl. Math. Comput.* **31**, 1–2 (2009), 369–384.
- [4] S. C. BARMAN, M. PAL, S. MONDAL. An optimal algorithm to find minimum k -hop dominating set of interval graphs. *Discrete Math. Algorithms Appl.* **11**, 2 (2019), 1950016, 18 pp, <https://doi.org/10.1142/S1793830919500162>.
- [5] P. BASUCHOWDHURI, S. MAJUMDER. Finding influential nodes in social networks using minimum k -hop dominating set. In: Applied Algorithms, ICAA 2014 (Eds P. Gupta, C. Zaroliagis) Lecture Notes in Computer Science, vol **8321**, Cham, Springer, https://doi.org/10.1007/978-3-319-04126-1_12.
- [6] D. A. BENSON, M. S. BOGUSKI, D. J. LIPMAN, J. OSTELL, B. F. OUELLETTE, B. A. RAPP, D. L. WHEELER. GenBank. *Nucleic Acids Research* **27**, 1 (1999) 12–17, <https://doi.org/10.1093/nar/27.1.10>.
- [7] S. BERCHTOLD, C. BÖHM, H. P. KRIEGEL. The pyramid-technique: Towards breaking the curse of dimensionality. In: Proc. Int. Conf. on Management of Data, ACM SIGMOD, Seattle, Washington, 1998, ACM SIGMOD Records, **27**, 2 (1998), <https://doi.org/10.1145/276305.276318>

- [8] S. BERCHTOLD, D. KEIM, H.-P. KRIEGEL. The X-tree: an index structure for high-dimensional data. VLDB '96, Proceedings of the 22th International Conference on Very Large Data Bases, (Eds T. M. Vijayaraman, A. P. Buchmann) September 1996, 28–39.
- [9] E. BERGER. Dynamic monopolies of constant size. Technical report, Los Alamos National Laboratories, 1999, arXiv:math/9911125.
- [10] A. A. BERTOSSI, A. GORI. Total domination and irredundance in weighted interval graphs. *SIAM J. Discrete Math.* **1**, 3 (1988), 317–327.
- [11] G. J. CHANG. Algorithmic aspect of domination in graphs. In: Handbook of Combinatorial Optimization, vol. 1, (Eds P. M. Pardalos, D.-Z. Du, R. L. Graham) New York, Springer, 2013, 221–282.
- [12] G. J. CHANG. The weighted independent domination problem is NP-complete for chordal graphs. *Discrete Appl. Math.* **143**, 1–3 (2004), 351–352.
- [13] M. S. CHANG. Weighted domination of cocomparability graphs. *Discrete Appl. Math.* **80**, 2–3 (1997), 135–148.
- [14] H. S. CHAO, F. R. HSU, R. C. T. LEE. An optimal algorithm for finding the minimum cardinality dominating set on permutation graphs. *Discrete Appl. Math.* **102**, 3 (2000), 159–173.
- [15] C. C.-Y. CHEN, S. K. DAS. Breadth-first traversal of trees and integer sorting in parallel. *Infom. Process. Lett.* **41**, 1 (1992), 39–49.
- [16] T. C. E. CHENG, L. Y. KANG, E. SHAN. A polynomial-time algorithm for the paired domination problem on permutation graphs. *Discrete Appl. Math.* **157**, 2 (2009), 262–271.
- [17] T. C. E. CHENG, L. Y. KANG, C. T. NG. Paired domination on interval and circular-arc graphs. *Discrete Appl. Math.* **155**, 16 (2007), 2077–2086.
- [18] L. CHEN, C. LU., Z. ZENG. A linear time algorithm for paired-domination problems in strongly chordal graphs, *Inform. Process. Lett.* **110**, 1 (2009), 20–23.
- [19] L. CHEN, C. LU, Z. ZENG. Labelling algorithm for paired-domination problems in block and interval graphs. *J. Comb. Optim.* **19**, 4 (2010), 457–470.

- [20] E. COCKAYNE, R. DAWES, S. HEDETNIEMI. Total domination in graphs. *Networks* **10**, 3 (1980), 211–219.
- [21] E. COCKAYNE, S. GOODMAN, S. HEDETNIEMI. A linear algorithm for the domination number of a tree. *Information Processing Lett.* **4**, 2 (1975), 41–44.
- [22] A. D’ATRI, M. MOSCARINI. Distance-hereditary graphs, Steiner trees, and connected domination. *SIAM J. Comput.* **17**, 3 (1988), 521–538.
- [23] C. F. DE JAENISCH. *Traite des Applications de l’Analyse Mathematique au Jeu des Echecs*. Petrograd, 1862.
- [24] E. D. DEMAINE, M. HAJJAGHAY, D. M. THILIKOS. Fixed parameter algorithm for (k, r) center in planar graphs and map graphs. *ACM Trans. Algorithms* **1**, 1 (2005), 1–16.
- [25] M. FARHADI JALALVAND, N. JAFARI RAD. On the complexity of k -step and k -hop dominating sets in graphs. *Math. Montisnigri* **40** (2017), 36–41.
- [26] M. C. GOLUMBIC. *Algorithmic graph theory and perfect graphs*. New York-London-Toronto, Ont., Academic Press, 1980.
- [27] T. W. HAYNES, S. T. HEDETNIEMI, P. J. SLATER. *Domination in Graphs: Advanced Topics*. Monographs and textbooks in pure and applied mathematics, vol. **209**, New York, Marcel Dekker, Inc., 1998.
- [28] T. W. HAYNES, S. T. HEDETNIEMI, P. J. SLATER. *Domination in Graphs: Selected Topics*. New York, Marcel Dekker, Inc., 1998.
- [29] T. W. HAYNES, S. T. HEDETNIEMI, P. J. SLATER. *Domination in Graphs: The Theory*. New York, Marcel Dekker, Inc., 1998.
- [30] T. W. HAYNES, S. T. HEDETNIEMI, P. J. SLATER. *Fundamentals of Domination in Graphs*. Monographs and Textbooks in Pure and Applied Mathematics, vol. **208**. New York, Marcel Dekker, Inc., 1998.
- [31] Haynes, T. W., Mynhardt, C. M. and van der Merwe, L. C., Total domination edge critical graphs, *Utilitas Math.* **54** (1998) 229–240.

- [32] Haynes, T. W., Mynhardt, C. M. and van der Merwe, L. C., Total domination edge critical graphs with maximum diameter, *Discussiones Math. Graph Theory* **21** (2001) 187–205.
- [33] T. W. HAYNES, P. J. SLATER. Paired-domination in graphs. *Networks* **32**, 3 (1998) 199–206.
- [34] S. T. HEDETNIEMI, R. C. LASKAR. Bibliography on domination in graphs and some basic definitions of domination parameters. *Discrete Math.* **86**, 1–3 (1990), 257–277.
- [35] S. T. HEDETNIEMI, R. C. LASKAR. Topics on Domination. Amsterdam, North-Holland, 1991.
- [36] M. A. HENNING, N. JAFARI RAD. On 2-step and hop dominating sets in graphs. *Graphs Combin.* **33**, 4 (2017), 913–927, <https://doi.org/10.1007/s00373-017-1789-0>.
- [37] R. W. HUNG. Linear time algorithm for the paired-domination problem in convex bipertite graphs. *Theory Comput. Syst.* **50**, 4 (2012), 721–738.
- [38] L. KANG, M. Y. SOHN, T. C. E. CHENG. Paired-domination in inflated graphs. *Theoret. Comput. Sci.* **320**, 2–3 (2004), 485–494.
- [39] S. KUNDU, S. MAJUMDER. A linear time algorithm for optimal k -hop dominating set of a tree. *Inform. Process. Lett.* **116**, 2 (2016), 197–202.
- [40] E. LAPPAS, S. D. NICOLOPOULOS, L. PALIOS. An $O(n)$ -time algorithm for the paired-domination problem on permutation graphs. Combinatorial algorithms *Lecture Notes in Comput. Sci.* vol. **5874**, 2009, 368–379.
- [41] C. S. LIAO, G. J. CHANG. Algorithmic aspect of k -tuple domination in graphs. *Taiwanese J. Math.*, **6**, 3 (2002) 415–420.
- [42] C. S. LIAO, G. J. CHANG. k -tuple domination in graphs. *Inform. Process. Lett.*, **87**, 1 (2003) 45–50.
- [43] C. LIU. Introduction to Combinatorial Mathematics. New York-Toronto, Ont.-London, McGraw-Hill Book Co., 1968.
- [44] C. L. LU, M.-T. KO, C. Y. TANG. Perfect edge domination and efficient edge domination in graphs. *Discrete Appl. Math.* **119**, 3 (2002) 227–250.

- [45] J. MCCOY, M. A. HENNING. Locating and paired-dominating sets in graphs. *Discrete Applied Math.* **157**, 15 (2009), 3268–3280.
- [46] S. MONDAL, M. PAL, T. K. PAL. An optimal algorithm to solve 2-neighbourhood covering problem on interval graphs. *Int. J. Comput. Math.* **79**, 2 (2002), 189–204.
- [47] J. H. MULLER, J. SPINRAD. Incremental modular decomposition. *J. Assoc. Comput. Mach.* **36**, 1 (1989) 1–19.
- [48] C. NATARAJAN, S. K. AYYASWAMY. Hop Domination in Graphs-II. *An. Ştiinţ. Univ. “Ovidius” Constanţa Ser. Mat.* **23**, 2 (2015), 187–199.
- [49] S. OLARIU. An optimal greedy heuristic to color interval graphs. *Inform. Process. Lett.* **37**, 1 (1991), 21–25.
- [50] M. PAL, G. P. BHATTACHARJEE. The parallel algorithms for determining edge-packing and efficient edge dominating sets in interval graphs. *Parallel Algorithms Appl.* **7**, 3–4 (1995), 193–207, <https://doi.org/10.1080/10637199508915531>.
- [51] B. S. PANDA, D. PRADHAN. Minimum paired-dominating set in chordal bipartite graphs and perfect elimination bipertite graphs. *J. Comb. Optim.* **26**, 4 (2013), 770–785 DOI:10.1007/s10878-012-9483-x.
- [52] A. PNUELI, A. LEMPEL, S. EVEN. Transitive orientation of graphs and identification of permutation graphs. *Canadian J. Math.*, **23** (1971), 160–175.
- [53] T. PRAMANIK, S. MONDAL, M. PAL. Minimum 2-tuple dominating set of an interval graphs. *Int. J. Comb.* **2011** (2011), Art. ID 389369, 14 pp.
- [54] H. QIAO, L. KANG, M. GARDEI, D.-Z. DU. Paired-domination of trees. *J. Global Optim.* **25**, 1 (2003) 43–54.
- [55] A. RANA, A. PAL, M. PAL. An efficient algorithm to solve the distance k -domination problem on permutation graphs, *J. Discrete Math. Sci. Cryptogr.* **19**, 2 (2016), 241–255.
- [56] E. SHAN, Y. DONG. The k -tuple twin domination in generalized de Bruijn and Kautz networks. *Comput. Math. Appl.* **63**, 1 (2012), 222–227.

- [57] P. J. SLATER. R -domination in graphs. *J. Assoc. Comput. Mach.* **23**, 2 (1976), 446–450.
- [58] D. P. SUMNER, P. BLITCH. Domination critical graphs. *J. Combin. Theory Ser. B* **34**, 1 (1983), 65–76.
- [59] R. TARJAN. Depth-first search and linear graph algorithm. *SIAM J. Comput.* **1**, 2 (1972), 146–160.
- [60] L. C. VAN DER MERWE, C. M. MYNHARDT, T. W., HAYNES. Criticality index of total domination. *Congr. Numer.* **131** (1998) 67–73.
- [61] M. YANNAKAKIS, F. GAVRIL. Edge dominating sets in graphs. *SIAM J. Appl. Math.* **38**, 3 (1980), 364–372.

Sambhu Charan Barman

Department of Mathematics

Shahid Matangini Hazra Govt. College for Women

Nimtouri, Purba Medinipur

721649 West Bengal, India

e-mail: barman.sambhu@gmail.com

Madhumangal Pal

Department of Applied Mathematics

with Oceanology and Computer Programming

Vidyasagar University

721102 Midnapore, India

e-mail: mmpalvu@gmail.com

Sukumar Mondal

Department Of Mathematics

Raja N. L. Khan Women'S College

Gope Palace

721102 Midnapore, India

e-mail: sm5971@rediffmail.com

Received January 3, 2020