Power Domination in Circular-arc Graphs[*][†]

Chung-Shou Liao[‡] D. T. Lee[§]

Department of Industrial Engineering and Engineering Management,
National Tsing Hua University, Hsinchu 300, Taiwan[‡]

Department of Computer Science and Engineering
National Chung Hsing University, Taichung 402, Taiwan[§]

Abstract

A set $S \subseteq V$ is a power dominating set (PDS) of a graph $G = (V, E)$ if every vertex and every edge in G can be observed based on the observation rules of power system monitoring. The power domination problem involves minimizing the cardinality of a PDS of a graph. We consider this combinatorial optimization problem and present a linear time algorithm for finding the minimum PDS of an interval graph if the interval ordering of the graph is provided. In addition, we show that the algorithm, which runs in $\Theta(n \log n)$ time, where n is the number of intervals, is asymptotically optimal if the interval ordering is not given. We also show that the results hold for the class of circular-arc graphs.

Keywords. domination; power domination; interval graphs; circular-arc graphs; algorithm.

1 Introduction

Continuous monitoring of power systems and observing all the states, such as the voltage magnitude of loads and the current phase measurements at branches, are important tasks for electric power companies [4, 28, 30, 34]. Placing phase measurement units (PMUs) at selected bus locations in a power system is an efficient way to monitor the system; for example, with real-time PMUs, fast transients can be tracked at high sampling rates. However, because of their high cost, the number of PMUs must be minimized without compromising their ability to monitor and observe the system. A power system is said to be *observed* if all the states can be determined by a set of PMUs according to the observation rules [4, 28]. A variety of heuristic approaches that approximate the minimum number of PMUs required have been developed in the last two decades [4, 12, 24, 25, 29].

The power system observation problem can be transformed into a graph-theoretic problem as follows [16]. Let $G = (V, E)$ be a graph representation of an electric power system, where a vertex represents an electric node (a substation bus that connects transmission branches, loads, and generators) and an edge represents a transmission branch that connects two electric nodes.

[*] An extended abstract of this paper was published in Proceedings of the 11th International Computing and Combinatorics Conference (COCOON), pp. 818-828, 2005.

[†] Supported in part by the National Science Council of Taiwan under Grants NSC100-2221-E-007-108-MY3, NSC99-2218-E-007-010, NSC98-2221-E-001-008-MY3, and NSC98-2221-E-001-007-MY3.

[‡] Email: csliao@ie.nthu.edu.tw

[§] Also with Dept. of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, and Institute of Information Science, Academia Sinica, Nankang, Taipei 115, Taiwan. Email: dtlee@iis.sinica.edu.tw

The problem of locating the smallest set of PMUs required to observe all the states of the power system is closely related to the famous vertex cover problem and the domination problem. A set $S \subseteq V$ is said to be a *power dominating set* (abbreviated as PDS) if every vertex and edge in G are observed by S according to the following PMU observation rules:

1. Any vertex where a PMU is placed and its incident edges are observed.
2. If one end vertex of an observed edge is observed, then the other end vertex is observed.
3. Any edge connecting two observed vertices is observed.
4. If a vertex is of degree $k > 1$, and $k - 1$ of its incident edges are observed, then all k incident edges are observed.

The minimum cardinality of a PDS of a graph G is called the *power domination number* of G, denoted by $\gamma_p(G)$. A set $D \subseteq V(G)$ is said to be a *dominating set* in a graph $G = (V, E)$ if every vertex in $V \setminus D$ is adjacent to at least one vertex in D. The cardinality of a minimum dominating set of a graph G is called the *domination number* of G, denoted by $\gamma(G)$. A *vertex cover* of a graph $G = (V, E)$ is a set $C \subseteq V(G)$ such that C contains at least one end vertex of every edge in $E(G)$. The cardinality of a minimum vertex cover of a graph G is denoted as $\beta(G)$. It is obvious that $1 \leq \gamma_p(G) \leq \gamma(G) \leq \beta(G)$ for any graph G. Figure 1 shows an example that highlights the differences between the power domination, domination, and vertex cover problems. Haynes *et al.* [16] considered the power domination problem as a variation of the domination problem and studied the relationship between them. They provided NP-completeness proofs for bipartite graphs and chordal graphs, and proposed a linear time algorithm for the power domination problem in trees. Guo *et al.* [15] showed that the power domination problem is also NP-complete for planar graphs, circle graphs, and split graphs, and it cannot be better approximated than the domination problem for general graphs. Liao and Lee [23] proposed a different NP-completeness proof for the power domination problem in split graphs. Subsequently, Aazami and Stilp [2] separated the approximation hardness of domination and power domination. They proved that, in contrast to the logarithmic threshold of the domination problem, the power domination problem cannot be approximated within the ratio $2^{\log^{1-\epsilon} n}$, unless NP $\subseteq$ DTIME$(n^{poly \log(n)})$. In addition, they proposed an $O(\sqrt{n})$-approximation algorithm for the power domination problem in planar graphs.

Some special classes of graphs have also been considered from an algorithmic point of view [3, 5, 10, 11, 19, 26, 27, 36]. Dorfling and Henning [11] and Pai *et al.* [26] determined the power domination number in grid graphs. Atkins *et al.* [3], Hon *et al.* [19], and Xu *et al.* [36] proposed linear time algorithms for the power domination problem in block graphs. A block graph is an intersection graph in which every maximal connected component (block) without a cut vertex is a clique. Dorbec *et al.* [10] considered the power domination number in product graphs, i.e., a variety of direct products of paths; and Pai *et al.* [27] investigated the restricted and fault-tolerant power domination problems and determined the power domination numbers in grid graphs. Moreover, Brueni and Heath [7, 8], and Zhao *et al.* [37] independently showed that the power domination number in a connected graph with $n \geq 3$ vertices is no larger than $\frac{n}{3}$, and characterized the extremal graphs that attain the upper bound.

In this paper, we consider the power domination problem in circular-arc graphs, one of the non-tree-type graph classes. Most works on the power domination problem in special classes of graphs have focused on tree-type graphs, such as trees and block graphs. Although the power domination problem in planar graphs has been investigated, the results are based on the *tree-width* property of planar graphs. The crucial difference between the power domination problem in tree-type and non-tree-type graphs is the number of interactions (called the *alternating break* in this paper) between

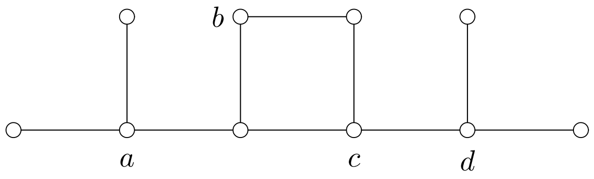

Figure 1: An instance of a graph G that illustrates the differences between the domination, vertex cover, and power domination problems. $D = \{a, b, d\}$ and $\gamma(G) = 3$, $C = \{a, b, c, d\}$ and $\beta(G) = 4$, but $S = \{a, d\}$ and $\gamma_p(G) = 2$.

the vertices in a PDS. that satisfy the fourth PMU observation rule The number of interactions among the vertices in a PDS may vary a great deal in a non-tree-type graph, and may be a critical issue when exploring the power domination problem.

The remainder of this paper is organized as follows. In Section 2 we introduce the notations and definitions used throughout the paper. In Section 3, we present a linear time algorithm for the power domination problem in interval graphs, which is a subclass of chordal graphs. In Section 4, we consider the same problem in proper circular-arc graphs and propose a linear time algorithm to solve it. In Section 5, we combine these two algorithmic strategies to extend our result to general circular-arc graphs. Section 6 contains some concluding remarks.

2 Notations and Definitions

A graph $H = (V_H, E_H)$ is a *subgraph* of $G = (V, E)$ if $V_H \subseteq V$ and $E_H \subseteq E$; and it is an *induced subgraph* of G, if for all u, $v \in V_H$, $\overline{uv} \in E_H$ if and only if $\overline{uv} \in E$. If $V_H = \{v_i, v_{i+1}, \ldots, v_k\}$, the induced subgraph $H = (V_H, E_H)$ is also written as $\{v_i, v_{i+1}, \ldots, v_k\}_G$. In the following, the subscript G, which denotes the underlying graph, is omitted without causing confusion. A vertex $w \in V$ is said to be a *neighbor* of, or *adjacent* to, a vertex $v \in V$ if $\overline{vw} \in E$. The *neighborhood* of a vertex $v \in V$ is $N_G(v) = \{w \in V \mid \overline{vw} \in E\}$; and the *closed neighborhood* of $v \in V$ is $N_G[v] = N_G(v) \cup \{v\}$. The *closed neighborhood* of a vertex set S, $N[S] = \bigcup_{s \in S} N[s]$. We define the *H-outdegree* of $v \in V_H$ of an induced subgraph $H = (V_H, E_H)$ of G as the number of vertices in $V \setminus V_H$ adjacent to v. The edge $\overline{vw} \in E$ connecting a vertex $v \in V_H$ and $w \notin V_H$ is called an *H-outgoing edge*. The observation rules for a vertex set $S = V^0$ in which PMUs are placed can be rewritten as follows. Similar arguments are presented in the literature [1, 2, 8, 15, 21, 31].

Induced Observation Rules

1. The sets of vertices and edges in the induced subgraph $K^1 = (V^1, E^1)$ of G are observed, where V^1 is the closed neighborhood of V^0. That is, $V^1 = N[V^0]$.

2. The sets of vertices and edges in the induced subgraph $K^i = (V^i, E^i)$ of G are observed, where $V^i = V^{i-1} \cup \{w \mid \overline{vw}$ is a V^{i-1}-*outgoing edge* and $v \in V^{i-1}$ is of V^{i-1}-*outdegree* $1\}$, $i \geq 2$.

Note that the new edge $\overline{vw} \in E$, where $v \in V^{i-1}$ is of V^{i-1}-*outdegree* 1, defined in the **Induced Observation Rule 2**, is exactly the same as that specified in the fourth PMU observation rule. The final graph $K^i = K^{i-1}$ for some $i > 0$ is called the *observed graph* of V^0, denoted by $\mathcal{G}_{V^0}$; and the size of $\mathcal{G}_{V^0}$, denoted by $|\mathcal{G}_{V^0}|$, is defined as the number of the vertices in V^i, i.e., $|\mathcal{G}_{V^0}|= |V^i|$. The set V^0 is a PDS of G if $\mathcal{G}_{V^0} = G$. The vertex set V^0 of the induced subgraph $K^0 = (V^0, E^0)$

of G is referred to as the *kernel*, and the vertices in the kernel are referred to as the kernel vertices. The subsequent vertex sets V^i, $i > 0$ are *derived kernels* of the i^{th} generation[1]. For ease of reference, the vertices in $V^i \setminus V^{i-1}$, $i > 0$ are called the i^{th} *generation descendants* (*i-descendants* for short) of those in V^0. Note that the Induced Observation Rules are equivalent to the original observation rules [1, 2, 8, 15, 21, 31]. In addition, given a graph $G = (V, E)$, the observed graph of some kernel V^0 can be computed in $O(|V| + |E|)$ time by the Induced Observation Rules [8].

Consider two kernels A and B and their respective observed graphs $\mathcal{G}_A$ and $\mathcal{G}_B$. The kernels are said to be *independent* if $|\mathcal{G}_{A\cup B}|$ is equal to $|\mathcal{G}_A \cup \mathcal{G}_B|$; otherwise, they are *dependent*, i.e., $|\mathcal{G}_{A\cup B}| > |\mathcal{G}_A \cup \mathcal{G}_B|$. The properties below follow from the Induced Observation Rules.

Property 2.1 *For two vertex sets U, W of a graph G, if $N[U] \subseteq N[W]$, then $\mathcal{G}_U \subseteq \mathcal{G}_W$. That is, the observed graph of kernel U is contained in the observed graph of kernel W if the closed neighborhood of U is a subset of that of W.*

Property 2.2 *Given a graph $G = (V, E)$, two kernels A, $B \subseteq V$ and their respective observed graphs $\mathcal{G}_A$, $\mathcal{G}_B$, kernel A and kernel B are dependent; that is, $|\mathcal{G}_{A\cup B}| > |\mathcal{G}_A \cup \mathcal{G}_B|$ if and only if there is a vertex $v \in \mathcal{G}_A$ of $\mathcal{G}_A$-outdegree k, such that among the k vertices adjacent to v, $k-1$ of them are in $\mathcal{G}_B$, or vice versa.*

3 Power Dominating Set for Interval Graphs

A graph G is called an *interval graph* if its vertices are in one-to-one correspondence with a set of intervals I of a linearly ordered set, such that two vertices are connected by an edge of G if and only if their corresponding intervals have nonempty intersections. We call I an *interval representation* of G. It has been shown that the class of interval graphs is a subclass of chordal graphs [14]. Interval graphs have been studied extensively in relation to the domination problem [9, 14, 17, 18], and most variations of the problem are solvable for this class of graphs. In the following discussion, we assume that an interval representation of the interval graph is available. Suppose $G = (V, E)$ is an interval graph, and its interval representation $\{I_i = [a_i, b_i] : 1 \leq i \leq n\}$ is indexed so that the right endpoints are sorted in order from left to right as follows: $b_1 \leq b_2 \leq \ldots \leq b_n$. The sequence of the corresponding vertices v_1, v_2, $\ldots$, v_n is called an *interval ordering* of G, and an interval graph can be recognized by seeking such an ordering in linear time [14]. The following interval ordering (IO) property is well-known [32]:

Property 3.1 *$G = (V, E)$ is an interval graph if and only if there exists an interval ordering v_1, v_2, $\ldots$, v_n such that the following condition holds.*

(IO) *If $i < j < k$ and $\overline{v_i v_k} \in E$, then $\overline{v_j v_k} \in E$.*

The above-mentioned interval representation, where I_i corresponds to v_i, possesses the interval ordering property. In this section, we present a linear time algorithm that can solve the power domination problem in an interval graph if an interval ordering of the graph is given. We assume that all the graphs discussed below are connected. First, we introduce the concept of a *gap*, which is used for choosing PMUs. Given an interval graph $G = (V, E)$ with an interval ordering $v_1, v_2, \ldots, v_n$, the corresponding intervals $I_i = [a_i, b_i]$ for every i satisfy $b_1 \leq b_2 \leq \ldots \leq b_n$. Without loss of generality, we assume that the left endpoint ordering of all the intervals is also given, i.e., $a'_1 \leq a'_2 \leq \ldots \leq a'_n$, where $a'_i \in \{a_1, \ldots, a_n\}$. For two successive right endpoints b_i and b_{i+1},

[1] Aazami and Stilp [1] independently presented a similar extension, called the ℓ-round PDS problem.

where $i \geq 1$, the pair (b_i, b_{i+1}) is called a *b-gap* if there is no vertex $v_k \neq v_{i+1}$ whose left endpoint a_k satisfies $b_i < a_k \leq b_{i+1}$. Similarly, an *a-gap* is a pair (a'_i, a'_{i+1}) of two successive left endpoints if there is no vertex $v_k \neq v'_i$ whose right endpoint b_k satisfies $a'_i \leq b_k < a'_{i+1}$. Both types of gap may contain more than two successive endpoints. Thus, for each b-gap, we define the first and last right endpoints, i.e., b_{f_i} and b_{ℓ_i} respectively; and for each a-gap, we define the first and last left endpoints, i.e., a'_{f_i} and a'_{ℓ_i} respectively. The set of all the endpoints on the real line can be marked with a sequence of labels with a and b representing the left and right endpoints respectively. When consecutive a's are grouped together to form an a-gap, we ignore any singleton b that joins its preceding a's to define an interval between consecutive a's. Similarly, when consecutive b's are grouped together to form a b-gap, we ignore any singleton a that joins its succeeding b's to define an interval between consecutive b's. As a result, we obtain a sequence of a-gaps and b-gaps that may be interleaved with singleton a's and b's. Figure 2 shows an example of a mixed sequence of a-gaps and b-gaps.

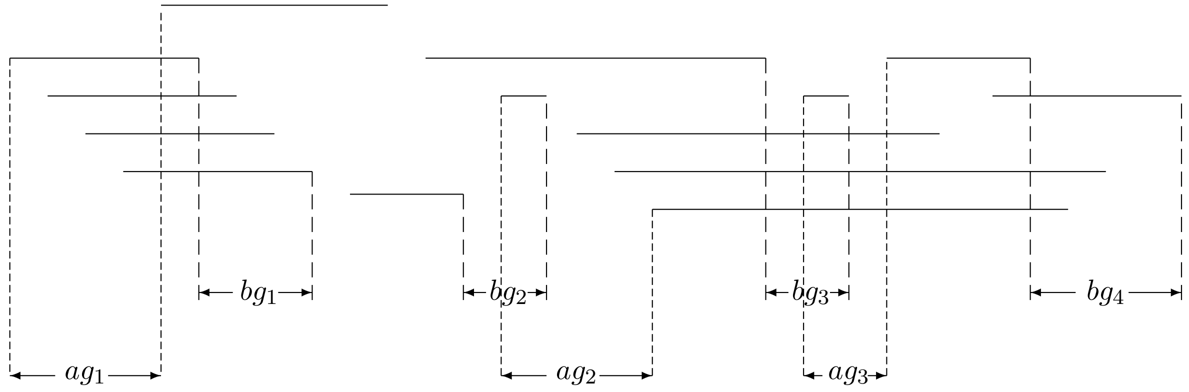

Figure 2: An example of a mixed sequence of a-gaps and b-gaps

Note that there may be an overlap of successive a-gaps and b-gaps. For instance, consider the subsequence a'_i, b_j, a'_{i+1}, b_{j+1}, where $a'_i = a_j$ and b_j define interval v_j, and $a'_{i+1} = a_{j+1}$ and b_{j+1} define interval v_{j+1}. By our definition, a'_i, b_j, a'_{i+1} form an a-gap, and b_j, a'_{i+1}, b_{j+1} form a b-gap. Similarly, there may also be an overlap of successive b-gaps and a-gaps (e.g., bg_3 and ag_3 in Figure 2). To form the above sequence of a-gaps and b-gaps interleaved with a's and b's, by pre-processing, we find all the a- and b-gaps, ag_1, ag_2, $\ldots$, ag_p and bg_1, bg_2, $\ldots$, bg_r respectively, where $ag_i = (a'_{f_i}, a'_{\ell_i})$ and $bg_j = (b_{f_j}, b_{\ell_j})$. In addition, the size of an a-gap ag_i (resp. a b-gap bg_j), denoted by $|ag_i|$ (resp. $|bg_j|$), is defined as the number of vertices that comprise the a-gap (resp. the b-gap), i.e., $|\{v'_{f_i}, v'_{f_i+1}, \ldots, v'_{\ell_i-1}, v'_{\ell_i}\}|$ (resp. $|\{v_{f_j}, v_{f_j+1}, \ldots, v_{\ell_j-1}, v_{\ell_j}\}|$). Obviously, $|ag_i|$, $|bg_j| \geq 2$ for each $1 \leq i \leq p$, $1 \leq j \leq r$. The notion of gaps plays an important role in the proposed algorithm, as we will show later.

Definition 3.2 *The b-gap $bg_j = (b_{f_j}, b_{\ell_j})$, $1 \leq j \leq r$ is a* blocking b-gap *of any vertex v_k, where the left endpoint of the interval corresponding to v_k lies to the right of b_{ℓ_j}, i.e., $a_k > b_{\ell_j}$. Similarly, the a-gap $ag_i = (a'_{f_i}, a'_{\ell_i})$, $1 \leq i \leq p$, is a* blocking a-gap *of any vertex v_k, where the right endpoint of the interval corresponding to v_k lies to the left of a'_{f_i}, i.e., $b_k < a'_{f_i}$.*

For ease of reference, we say that the b-gap $bg_j = (b_{f_j}, b_{\ell_j})$, $1 \leq j \leq r$ is a *left* blocking gap of any vertex v_k that lies to the *right* of b_{ℓ_j}, i.e., $a_k > b_{\ell_j}$; and that the a-gap $ag_i = (a'_{f_i}, a'_{\ell_i})$,

$1 \le i \le p$, is a *right* blocking gap of any vertex v_k that lies to the *left* of a'_{f_i}, i.e., $b_k < a'_{f_i}$.

Lemma 3.3 (blocking gap lemma) *Let* $bg_j = (b_{f_j}, b_{\ell_j})$, $1 \le j \le r$ *be a* left blocking b-gap *of vertex* v_k*, and let* $ag_i = (a'_{f_i}, a'_{\ell_i})$, $1 \le i \le p$, *be a* right blocking a-gap *of vertex* v_k*. Then, each vertex* v_u *with* $b_u \le b_{\ell_j}$ *cannot belong to the observed graph of* $\{v_k\}$*, and each vertex* v_u *with* $a_u \ge a'_{f_i}$ *cannot belong to the observed graph of* $\{v_k\}$*.*

Proof. Let vertices v_{f_j}, v_{f_j+1}, $\ldots$, v_{ℓ_j-1}, v_{ℓ_j} be the successive vertices that define the b-gap $bg_j = (b_{f_j}, b_{\ell_j})$, $1 \le j \le r$. That is, there is no vertex $v_w \ne v_{f_j}$, v_{f_j+1}, $\ldots$, v_{ℓ_j-1}, v_{ℓ_j} such that $b_{f_j} < a_w \le b_{\ell_j}$. Since v_k, whose $a_k > b_{\ell_j}$, is not adjacent to vertices that define bg_j, if there were a vertex v_u with $b_u \le b_{\ell_j}$ that belonged to $\mathcal{G}_{\{v_k\}}$, it would be included in $\mathcal{G}_{\{v_k\}}$ because of the Induced Observation Rule 2. Let V^0 be $\{v_k\}$ and let $K^t = (V^t, E^t)$ be the t^{th} generation induced subgraph of the observed graph $\mathcal{G}_{V^0}$ for some t. Clearly, the right endpoint of each vertex $v_w \in N[v_k] = V^1$, $a_k > b_{\ell_j}$, must lie to the right of a_k. Since the b-gap, bg_j, is defined as the successive vertices v_{f_j}, v_{f_j+1}, $\ldots$, v_{ℓ_j-1}, v_{ℓ_j}, each vertex $v_w \in V^t$, where $t \ge 1$, adjacent to some vertex v_u with $b_u \le b_{\ell_j}$ must be adjacent to all of the successive vertices that define bg_j. That is, there are at least two V^t-outgoing edges from $v_w \in V^t$, namely, $\overline{v_w v_{f_j}}$, and $\overline{v_w v_{\ell_j}}$. Thus, v_u with $b_u \le b_{\ell_j}$ cannot be included in the observed graph $\mathcal{G}_{\{v_k\}}$ by the Induced Observation Rule 2. Similarly, we can prove the latter statement that each vertex v_u with $a_u \ge a'_{f_i}$ cannot belong to $\mathcal{G}_{\{v_k\}}$ whose kernel vertex v_k satisfies $b_k < a'_{f_i}$. □

For each vertex v_k among all left blocking b-gaps $bg_j = (b_{f_j}, b_{\ell_j})$ for some j, $1 \le j \le r$, the one with the largest b_{ℓ_j} that is smaller than a_k is referred to as *the left* blocking b-gap of v_k. Similarly, for each vertex v_k among all right blocking a-gaps $ag_i = (a'_{f_i}, a'_{\ell_i})$ for some i, $1 \le i \le p$, the one with the smallest a'_{f_i} that is greater than b_k is referred to as *the right* blocking a-gap of v_k.

Associated with a b-gap $bg_i = (b_{f_i}, b_{\ell_i})$, $1 \le i \le r$, we have a *PMU candidate* (*candidate* for short), v_{c_i}, which is the vertex adjacent to v_{f_i} and whose corresponding interval has the maximum right endpoint b_{c_i} among those with this property. Recall that the vertex v_{c_i} corresponds to an interval $[a_{c_i}, b_{c_i}]$ and all the vertices $v_{f_i}, v_{f_i+1}, \ldots, v_{\ell_i-1}, v_{\ell_i}$ that define bg_i are in $N[v_{c_i}]$. Therefore, the vertices $v_{f_i}, v_{f_i+1}, \ldots, v_{\ell_i-1}, v_{\ell_i}$ that define the b-gap bg_i are all contained in $\mathcal{G}_{\{v_{c_i}\}}$. We assume that the PMU candidate v_{c_r} associated with the last b-gap $bg_r = (b_{f_r}, b_{\ell_r})$, where $b_{\ell_r} = b_n$, is v_n. The next two lemmas follow immediately.

Lemma 3.4 (backward observation lemma) *Let the b-gap* $bg_k = (b_{f_k}, b_{\ell_k})$ *be the left blocking gap of a candidate* v_{c_i} *for some* i, $1 \le i \le r$*. The induced subgraph* $\{v_{\ell_k+1}, \ldots, v_{f_i}, \ldots, v_{\ell_i}, \ldots, v_{c_i}\}$ *is contained in the observed graph* $\mathcal{G}_{\{v_{c_i}\}}$ *of the kernel* $\{v_{c_i}\}$*. We call this generation of observed vertices and edges a backward observation from the kernel* $\{v_{c_i}\}$*. The backward observation from* $\{v_{c_i}\}$ *stops at the left blocking gap* bg_k *of* v_{c_i}*.*

Proof. Let $V^0 = \{v_{c_i}\}$ and consider the vertices v_{ℓ_k+1}, $\ldots$, v_{f_i}, $\ldots$, v_{ℓ_i}, $\ldots$, v_{c_i}. The vertices in $N[v_{c_i}] = V^1$, which definitely includes $v_{f_i}, \ldots, v_{\ell_i}, \ldots, v_{c_i}$, are 1-descendants of V^0. Excluding the vertices in V^1, we consider the rest of the vertices in descending order of their right endpoints. Let v_u be the first vertex (of the maximum right endpoint) that is not one of the vertices that define bg_i. If $v_u = v_{\ell_k}$, we are done. Otherwise, there must be a vertex v_w, $v_w \ne v_u$, in V^1 such that $b_{u-1} < a_w \le b_u$ (where v_{u-1} may be v_{ℓ_k}). This is because there may be a sequence of interleaving singleton a's and b's between two consecutive b-gaps. We know that $\overline{v_u v_w}$ is a V^1-outgoing edge and v_w is of V^1-outdegree 1. Therefore, v_u is the 2-descendant of V^0. By repeating this argument, we can show that the vertex v_{u-t} is the $(t+2)$-descendant of V^0, where $0 \le t < u - \ell_k$. Thus, the induced subgraph $\{v_{\ell_k+1}, \ldots, v_{f_i}, \ldots, v_{\ell_i}, \ldots, v_{c_i}\}$ is contained in $\mathcal{G}_{\{v_{c_i}\}}$. Finally, similar to the proof of Lemma 3.3, v_{f_k}, ..., v_{ℓ_k} are not adjacent to v_{c_i}; hence, they are not in $N[v_{c_i}]$. Since the vertices define a b-gap, they are not t-descendant vertices of the kernel $\{v_{c_i}\}$ for any $t > 0$. The left blocking b-gap bg_k stops the backward observation from $\{v_{c_i}\}$. □.

Lemma 3.5 (forward observation lemma) *Suppose that the a-gap $ag_j = (a'_{f_j}, a'_{\ell_j})$ is the right blocking gap of a candidate v_{c_i} for some i, $1 \leq i \leq r$. Let v'_u be the vertex to the immediate right of a_{c_i}, i.e., $a'_u > a_{c_i}$ and let there be no other left endpoint between them. The induced subgraph $\{v_{c_i}, v'_u, v'_{u+1}, \ldots, v'_{f_j-1}\}$ is contained in the observed graph $\mathcal{G}_{\{v_{c_i}\}}$ of the kernel $\{v_{c_i}\}$. We call this generation of observed vertices and edges a forward observation from the kernel $\{v_{c_i}\}$. The forward observation from $\{v_{c_i}\}$ stops at the right blocking gap ag_j of v_{c_i}.*

Proof. The proof is similar to that of Lemma 3.4 and is therefore omitted. □.

The following lemma illustrates the role of PMU candidates on which we base our algorithm.

Lemma 3.6 *Given an interval graph $G = (V, E)$, there exists an optimal PDS S for G consisting exclusively of PMU candidates associated with b-gaps, i.e., $S \subseteq \{v_{c_1}, v_{c_2}, \ldots, v_{c_r}\}$.*

Proof. Suppose there is an optimal PDS S for G in which not all vertices are PMU candidates. Let v_{c_1}, v_{c_2}, ..., v_{c_r} be the sequence of all PMU candidates of G in ascending order, and consider a vertex $v_u \in S \setminus \{v_{c_1}, v_{c_2}, \ldots, v_{c_r}\}$. We select the first b-gap bg_k whose associated PMU candidate $v_{c_k} = [a_{c_k}, b_{c_k}]$ satisfies the condition that b_{c_k} is greater than b_u. Then, we consider the induced subgraph $G' \subseteq G$ of vertices whose right endpoints are bigger than b_{c_k}. We have $N_{G'}[v_u] \subseteq N_{G'}[v_{c_k}]$ and $\mathcal{G}_{\{v_u\}} \cap G' \subseteq \mathcal{G}_{\{v_{c_k}\}} \cap G'$ by Properties 2.1 and 3.1. Now, suppose $bg_j = (b_{f_j}, b_{\ell_j})$ is the left blocking gap of v_{c_k}. Then, v_u cannot be adjacent to any of the vertices that define bg_j because $b_{c_j} < b_u$ by our selection. Thus, by Lemma 3.3, the b-gap bg_j is the left blocking gap of both v_u and v_{c_k}. In addition, by Lemma 3.4, the induced subgraph $\{v_{\ell_j+1}, \ldots, v_{c_k}\}$ is contained in the observed graph $\mathcal{G}_{\{v_{c_k}\}}$. This implies that $\mathcal{G}_{\{v_u\}} \subseteq \mathcal{G}_{\{v_{c_k}\}}$. If, on the other hand, the left blocking b-gap bg_j does not exist, that is, $v_{c_k} = v_{c_1}$, then, by Lemma 3.4, the induced subgraph $\{v_1, v_2, \ldots, v_{c_k} = v_{c_1}\}$ is contained in the observed graph $\mathcal{G}_{\{v_{c_k}\}}$. This also implies that $\mathcal{G}_{\{v_u\}} \subseteq \mathcal{G}_{\{v_{c_k}\}}$. Then, $v_u \in S$ can be replaced by v_{c_k} to obtain a new $S = S \setminus \{v_u\} \cup \{v_{c_k}\}$. By repeating this replacement argument, we can derive an optimal PDS S consisting exclusively of PMU candidates. □

We explain the key concept behind our solution to the power domination problem in connected interval graphs. The PMU candidate v_{c_1} associated with the first b-gap bg_1 must be chosen first because of Lemmas 3.3 and 3.6. If we were to choose $v_{c_j} \neq v_{c_1}$ with $j > 1$, then bg_1 would be the left blocking gap of v_{c_j} and v_1, in particular, would not belong to $\mathcal{G}_{\{v_{c_j}\}}$. Thus, v_{c_1} must be chosen and the forward observation from $\{v_{c_1}\}$ will proceed until the right blocking gap of v_{c_1} is reached. We consider the choice of the next PMU candidate v_{c_i} in a greedy manner such that all the vertices between v_{c_1} and v_{c_i} belong to $\mathcal{G}_{\{v_{c_1}, v_{c_i}\}}$ and the index i is as large as possible. If we can choose the next candidate v_{c_i} correctly, then, by repeating the same strategy we claim that we will find the optimal PDS. To choose the candidate, we need to consider the necessary and sufficient conditions for the kernels $\{v_{c_i}\}$ and $\{v_{c_k}\}$ to be *complete*. We say that two kernels $\{v_{c_i}\}$ and $\{v_{c_k}\}, c_i < c_k$ are *complete* if all the vertices between v_{c_i} and v_{c_k} belong to $\mathcal{G}_{\{v_{c_i}, v_{c_k}\}}$; otherwise, they are *incomplete*. In addition, we say $\{v_{c_k}\}$ is *maximally complete* with respect to $\{v_{c_i}\}$ if they are complete and $c_k - c_i$ is the maximum, i.e., we cannot find a vertex v_{c_j}, $c_j > c_k$, such that $\{v_{c_i}\}$ and $\{v_{c_j}\}$ are complete.

Next, we define some terms. The *essential spot* of an a-gap $ag_i = (a'_{f_i}, a'_{\ell_i})$, denoted by $ess(ag_i)$, is the second smallest right endpoint of the vertices v'_{f_i}, v'_{f_i+1}, ..., v'_{ℓ_i} that define the a-gap. In addition, we say a vertex set S *breaks* a b-gap $bg_j = (b_{f_j}, b_{\ell_j})$ (resp. an a-gap $ag_i = (a'_{f_i}, a'_{\ell_i})$) if at least $|bg_j| - 1$ vertices among $v_{f_j}, \ldots, v_{\ell_j}$ (resp. at least $|ag_i| - 1$ vertices among $v'_{f_j}, \ldots, v'_{\ell_j}$) belong to the observed graph $\mathcal{G}_S$. Obviously, if the essential spot $ess(ag_i)$ of some a-gap ag_i lies to the immediate right of some b-gap bg_k, i.e., $ess(ag_i) > b_{\ell_k}$, then, by Lemma 3.4, the kernel $\{v_{c_{k+1}}\}$ breaks the a-gap ag_i. Similarly, we define the *essential spot* of a b-gap $bg_k = (b_{f_k}, b_{\ell_k})$, denoted by $ess(bg_k)$, as the second largest left endpoint of the vertices $v_{f_k}, v_{f_{k+1}}, \ldots, v_{\ell_k}$ that define the b-gap. Thus, if the essential spot $ess(bg_k)$ of some b-gap bg_k lies to the immediate left of an a-gap ag_i, i.e., $ess(bg_k) < a'_{f_i}$, then the b-gap bg_k will be *broken* by the kernel $\{v_c\}$, where v_c lies to the left of a'_{f_i}.

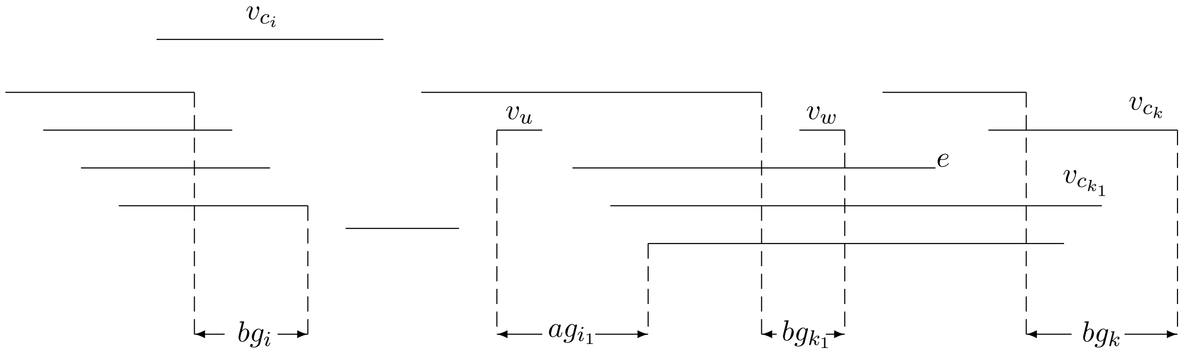

Figure 3: An instance in which the kernels $\{v_{c_i}\}$ and $\{v_{c_k}\}$ are complete; the right endpoint e is the essential spot of ag_{i_1}.

Figure 3 shows an example in which the kernels $\{v_{c_i}\}$ and $\{v_{c_k}\}$ are complete. First, we consider the observed graph $\mathcal{G}_{\{v_{c_i}, v_{c_k}\}}$ and explain why the kernels are complete. Note that the forward observation from the kernel $\{v_{c_i}\}$ stops at the a-gap ag_{i_1}. However, the backward observation from the kernel $\{v_{c_k}\}$ breaks the a-gap ag_{i_1} because the essential spot $ess(ag_{i_1})$ lies to the right of the b-gap bg_{k_1}, which is the left blocking b-gap of v_{c_k}. As a result, the forward observation from $\{v_{c_i}\}$ can continue in ascending order and eventually join the backward observation from $\{v_{c_k}\}$ so that all the vertices between v_{c_i} and v_{c_k} will be in $\mathcal{G}_{\{v_{c_i}, v_{c_k}\}}$. Similarly, the forward observation from $\{v_{c_i}\}$ also breaks the left blocking b-gap bg_{k_1} of v_{c_k} such that the backward observation from $\{v_{c_k}\}$ can continue in descending order. Joining these forward and backward observations ensures that all the vertices between v_{c_i} and v_{c_k} will be in $\mathcal{G}_{\{v_{c_i}, v_{c_k}\}}$. These two scenarios can occur independently of each other. The kernels $\{v_{c_i}\}$ and $\{v_{c_k}\}$ will be complete under either scenario.

Let us re-examine the above example and consider the union of the observed graphs $\mathcal{G}_{\{v_{c_i}\}}$ and $\mathcal{G}_{\{v_{c_k}\}}$. Recall that the forward observation from the kernel $\{v_{c_i}\}$ stops at the a-gap ag_{i_1} and the backward observation from the kernel $\{v_{c_k}\}$ stops at the b-gap bg_{k_1}. Since there exist vertices v_u, v_w that do not belong to $\mathcal{G}_{\{v_{c_i}\}} \cup \mathcal{G}_{\{v_{c_k}\}}$, it is easy to see that kernels $\{v_{c_i}\}$ and $\{v_{c_k}\}$ are dependent. Thus, we say the kernels $\{v_{c_i}\}$ and $\{v_{c_k}\}$ are *dependent complete*. On the other hand, if all the vertices between v_{c_i} and v_{c_k} belong to $\mathcal{G}_{\{v_{c_i}\}} \cup \mathcal{G}_{\{v_{c_k}\}}$, i.e., $\mathcal{G}_{\{v_{c_i}\}} \cup \mathcal{G}_{\{v_{c_k}\}} = \mathcal{G}_{\{v_{c_i}, v_{c_k}\}}$, we say the kernels $\{v_{c_i}\}$ and $\{v_{c_k}\}$ are *independent complete*.

As shown in Figure 4, the process of an *alternating break* of the blocking gaps (the formal definition will be given later) and the propagation of forward or backward observations might not ensure that all the vertices between v_{c_i} and v_{c_k} are in $\mathcal{G}_{\{v_{c_i}, v_{c_k}\}}$; thus, kernels $\{v_{c_i}\}$ and $\{v_{c_k}\}$ are incomplete. The figure shows an example in which $\{v_{c_i}\}$ and $\{v_{c_k}\}$ are incomplete.

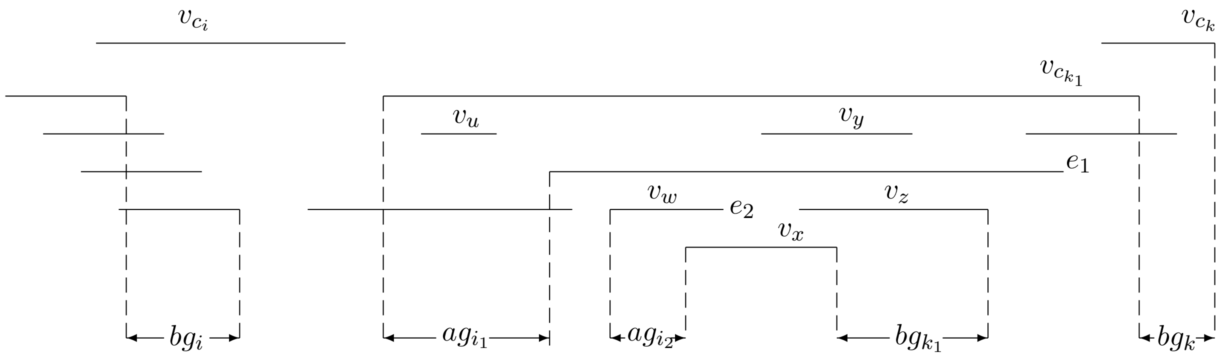

Figure 4: An instance that shows $\{v_{c_i}\}$ and $\{v_{c_k}\}$ are incomplete, where e_1 (resp. e_2) is the essential spot of ag_{i_1} (resp. ag_{i_2}).

Consider the union of the observed graphs $\mathcal{G}_{\{v_{c_i}\}}$ and $\mathcal{G}_{\{v_{c_k}\}}$. As before, the forward observation from the kernel $\{v_{c_i}\}$ in Figure 4 stops at the a-gap ag_{i_1}, and the backward observation from the kernel $\{v_{c_k}\}$ stops at the b-gap bg_{k_1}. On the other hand, consider the observed graphs $\mathcal{G}_{\{v_{c_i},v_{c_k}\}}$. Note that the essential spot of ag_{i_1}, $ess(ag_{i_1})$, lies to the right of the left blocking b-gap bg_{k_1}, and the backward observation from $\{v_{c_k}\}$ breaks the a-gap ag_{i_1}, thereby allowing the forward observation from $\{v_{c_i}\}$ to continue in ascending order of left endpoints and stop at the a-gap ag_{i_2}. However, since the forward observation from $\{v_{c_i}\}$ does not break the left blocking b-gap bg_{k_1} of v_{c_k}, the forward observation and backward observation will stop at the a-gap ag_{i_2} and b-gap bg_{k_1} respectively. As a result, the vertices v_w, v_x, v_y, v_z do not belong to the observed graph $\mathcal{G}_{\{v_{c_i},v_{c_k}\}}$, so $\{v_{c_i}\}$ and $\{v_{c_k}\}$ are incomplete.

Having explained the notion of completeness and incompleteness, we consider the process of an *alternating break* of the blocking gaps. An *alternating break* of the blocking gaps is defined by the next procedure Alternate Break, which is similar to the previous two examples. To facilitate the manipulation of broken gaps, we use two *doubly-linked lists* $A[\ ]$ and $B[\ ]$ to store the respective a-gap and b-gap sequences. The steps of the procedure Alternate Break are as follows.

The procedure Alternate Break shows the iterative propagation of the forward observation from $\{v_{c_i}\}$ and the backward observation from $\{v_{c_k}\}$ alternately. We call this process an *alternating break* executed by two kernels $\{v_{c_i}\}$ and $\{v_{c_k}\}$. Based on this definition, we characterize the notion of completeness in the following lemma.

Lemma 3.7 *Given a connected interval graph $G = (V, E)$, kernels $\{v_{c_i}\}$ and $\{v_{c_k}\}$, $c_i < c_k$, are complete, i.e., all the vertices between v_{c_i} and v_{c_k} belong to the observed graph $\mathcal{G}_{\{v_{c_i},v_{c_k}\}}$, if and only if no a-gap or a b-gap between v_{c_i} and v_{c_k} remains unbroken by the alternating breaks of the two kernels $\{v_{c_i}\}$ and $\{v_{c_k}\}$; that is, the procedure* Alternate Break *returns Success.*

Proof. If all the a-gaps and b-gaps between v_{c_i} and v_{c_k} are broken by the alternating breaks of the two kernels $\{v_{c_i}\}$ and $\{v_{c_k}\}$, then, by Lemmas 3.4 and 3.5, all the vertices between v_{c_i} and v_{c_k} belong to the observed graph $\mathcal{G}_{\{v_{c_i},v_{c_k}\}}$, i.e., $\{v_{c_i}\}$ and $\{v_{c_k}\}$ are complete. Conversely, assume the kernels $\{v_{c_i}\}$ and $\{v_{c_k}\}$ with $c_i < c_k$ are complete, and that there exist a-gaps or b-gaps between v_{c_i} and v_{c_k} that remain unbroken by the alternating breaks of the two kernels $\{v_{c_i}\}$ and $\{v_{c_k}\}$. Consider the case where there are only a-gaps between v_{c_i} and v_{c_k}. This is impossible since the

```
Procedure Alternate Break($v_{c_i}$, $v_{c_k}$)

  1. Initialize $j = 0$;        /* $j$ is the number of alternating breaks. */

  2. do {

    2-1. $j = j + 1$;
    2-2. Label each $b$-gap $bg$ in $B[\,]$ as "broken" if the forward observation from the kernel
         $\{v_{c_i}\}$ propagates through $ess(bg)$ and stops at $ag_{i_j}$;   /* $ess(bg)$ lies to the left of $ag_{i_j}$.*/
    2-3. if (there is no $a$-gap $ag_{i_j}$)
             return Success;
    2-4. Label each $a$-gap $ag$ in $A[\,]$ as "broken" if the backward observation from the kernel
         $\{v_{c_k}\}$ propagates through $ess(ag)$ and stops at $bg_{k_j}$;   /* $ess(ag)$ lies to the right of $bg_{k_j}$.*/

     } while($bg_{k_j}$ exists and the backward observation from $\{v_{c_k}\}$ breaks $ag_{i_j}$)

  3. if(there is a pair of gaps, an $a$-gap $ag_{i_j}$ and a $b$-gap $bg_{k_j}$ between $v_{c_i}$ and $v_{c_k}$)
         return Failure;
     else
         return Success;
```

backward observation from $\{v_{c_k}\}$ breaks all the a-gaps. Similarly, assume that there exist only b-gaps between v_{c_i} and v_{c_k}. This is also impossible because the forward observation from $\{v_{c_i}\}$ breaks all the b-gaps. Lastly, if there exists a pair consisting of an a-gap and a b-gap between v_{c_i} and v_{c_k} that remains unbroken by the alternating breaks of the two kernels $\{v_{c_i}\}$ and $\{v_{c_k}\}$, then the a-gap stops the forward observation from $\{v_{c_i}\}$ and the backward observation from $\{v_{c_k}\}$ cannot break it. Meanwhile, the b-gap stops the backward observation from $\{v_{c_k}\}$ and the forward observation from $\{v_{c_i}\}$ cannot break it. It is impossible for $\{v_{c_i}\}$ and $\{v_{c_k}\}$ to be complete, which contradicts our assumption. □

Corollary 3.8 *Given a connected interval graph $G = (V, E)$, the kernels $\{v_{c_i}\}$ and $\{v_{c_k}\}$, $c_i < c_k$, are incomplete if and only if there exists at least one pair consisting of an a-gap and a b-gap between v_{c_i} and v_{c_k} that are unbroken by the alternating breaks of the kernels $\{v_{c_i}\}$ and $\{v_{c_k}\}$.*

Recall that $\{v_{c_i}\}$ and $\{v_{c_k}\}$ in Figure 4 are incomplete. This means that the index c_k of v_{c_k} is too big to cooperate with v_{c_i}, and all the vertices between them cannot belong to the observed graph $\mathcal{G}_{\{v_{c_i}, v_{c_k}\}}$. The reason is that there exists one pair consisting of an a-gap ag_{i_2} stopping the forward observation from $\{v_{c_i}\}$ and a b-gap bg_{k_1} stopping the backward observation from $\{v_{c_k}\}$. Another possible choice of a PMU candidate being complete with $\{v_{c_i}\}$ is v_{c_j}, which is associated with bg_j. This candidate is chosen to break the a-gap, ag_{i_2}; that is, $ess(ag_{i_2})$ lies to the immediate left of, or belongs to, the b-gap bg_j. Obviously, $j \leq k_1$; otherwise, the backward observation from $\{v_{c_k}\}$ stopped by the b-gap bg_{k_1} could also break the a-gap ag_{i_2}, and the alternating break executed by $\{v_{c_i}\}$ and $\{v_{c_k}\}$ would repeat. In Figure 4, $v_{c_{k_1}}$ is a new possible choice that is complete with $\{v_{c_i}\}$. We use Lemma 3.7 to check the completeness condition between $\{v_{c_i}\}$ and the new possible choice of kernel $\{v_{c_j}\}$ again. If $\{v_{c_i}\}$ and $\{v_{c_j}\}$ satisfy the completeness condition, v_{c_j} is a feasible choice; otherwise, we choose the next possible candidate of smaller index and check the completeness condition described above until a feasible choice is found.

In order to quickly find a pair of PMU candidates that are maximally complete, for each a-gap ag, we maintain a *forward* pointer $a2b(ag)$ to the b-gap, bg_k, where $ess(ag)$ lies to the immediate left of, or belongs to, the b-gap bg_k. Similarly, for each b-gap bg, we maintain a *back* pointer $b2a(bg)$ to the a-gap, ag_i, where $ess(bg)$ lies to the immediate right of, or belongs to, the a-gap ag_i. As part of the preprocessing step, for each endpoint (a_i or b_i), we also have a pointer to the next a-gap on its immediate right, and the previous b-gap on its immediate left. Consider a PMU candidate v_{c_i} and suppose the forward observation from $\{v_{c_i}\}$ is blocked by an a-gap ag. To break this a-gap, we need to break the b-gap bg_k pointed to by $a2b(ag)$. Thus, v_{c_k}, the PMU candidate associated with bg_k, may be a possible candidate that may be complete with $\{v_{c_i}\}$. To determine if $\{v_{c_k}\}$ is complete with $\{v_{c_i}\}$, we need to consider the backward observation from $\{v_{c_k}\}$ and find its left blocking b-gap bg. Similarly, if the a-gap ag_j pointed to by $b2a(bg)$ is broken, then the associated bg is no longer a b-gap, and the backward observation from $\{v_{c_k}\}$ can propagate and break the associated a-gaps as well.

Moreover, if some b-gap bg_k is broken by a forward observation, then for each of its associated a-gaps whose forward pointers $a2b(\ )$ point to bg_k, say ag_i, we need to determine the first unbroken b-gap $bg_{k'}$ subsequent to bg_k, i.e., $k' > k$. If the candidate $v_{c_{k'}}$, which is associated with $bg_{k'}$ can be found, then it will break the a-gap ag_i and may be maximally complete with respect to the candidates preceding ag_i. We can consider this pointer-update problem as a *disjoint set union-find* problem. Initially every b-gap bg forms a subset consisting of a-gaps whose forward pointers point to bg; that is, we partition all the a-gaps into r subsets, where r is the number of b-gaps dependent on $a2b(\ )$. If some b-gap bg_k is broken by the forward observation, the subset bg_k will be merged into the next subset bg_{k+1}. This is called a *union* operation. If we need to find a possible candidate v_c maximally in order to break some blocking a-gap ag via its forward pointer $a2b(ag)$, then we have to output the index of the b-gap subset containing ag. This is called a *find* operation. The best known algorithm for the general disjoint set union-find problem (m operations on n elements) runs in $O(m\alpha(m+n)+n)$ time [33], where α is the inverse of Ackermann's function, which is slightly more than linear time. However, our pointer-update problem is actually a special case of the disjoint set union-find problem defined by Gabow and Tarjan [13]. We can use the *static tree set union* and *table look-up* techniques proposed in [13] on a two-level data structure of this static tree (*microsets* and *macrosets*) to solve our pointer-update problem in linear time. The key point of the *static tree set union* is that the structure of the union operations represented by a union tree is known in advance. Accordingly, the static union tree can be constructed by linking bg_k and bg_{k+1}, $1 \le k < r$, in our case. Since the static union tree is known in advance, the find operations on smaller *microsets* can be pre-computed via their associated lookup tables. Thus, our pointer-update problem can be solved in linear time.

Based on the above discussion, we present a linear time algorithm MPDI to solve the power domination problem in a given connected interval graph. The high level idea of the MPDI algorithm can be described as follows. As mentioned earlier, in the initial step, we have to select v_{c_1} as the first PMU candidate and let $v_c = v_{c_1}$. Then, we choose the next possible PMU candidate that is maximally complete with respect to $\{v_c\}$. We claim that each selection of the next PMU candidate v_c^* will be as large as possible, and that the procedure Alternate Break will check if the sufficient completeness condition between $\{v_c\}$ and $\{v_c^*\}$ holds. If the condition holds, we select the next PMU candidate that is maximally complete with respect to $\{v_c^*\}$ and repeat the argument; otherwise, we repeat the maximal selection of a possible PMU candidate until we find the candidate that is maximally complete with respect to $\{v_c\}$.

Lemma 3.9 *The selection of possible candidates in Algorithm* MPDI *finds the maximally complete candidate v_c^* with respect to the kernel $\{v_c\}$ after the procedure* Alternate Break *returns Success.*

Algorithm 1: **MPDI.** Find a minimum PDS of a connected interval graph.

```
Input. A connected interval graph $G = (V, E)$ with an interval ordering $v_1$, $v_2$, $\ldots$, $v_n$. A
    global doubly-linked list $A[\ ]$ consisting of all the $a$-gaps $ag_1$, $ag_2$, $\ldots$, $ag_p$ and their
    essential spots $ess(ag_i)$, $1 \leq i \leq p$, and a global doubly-linked list $B[\ ]$ consisting of all
    the $b$-gaps $bg_1$, $bg_2$, $\ldots$, $bg_r$ and their PMU candidates $v_{c_1}$, $v_{c_2}$, $\ldots$, $v_{c_r}$.

Output. A minimum PDS $S$ of $G$.

Method.

        1. Let $S = \{v_{c_1}\}$ and $v_c = v_{c_1}$;

        2. do {

            2-1. Find the right blocking $a$-gap $ag_i$ of $v_c$; if there is no $a$-gap, return $S$;

            2-2. Select a possible candidate $v_c^* = v_{c_k}$ associated with the $b$-gap $bg_k$, where $bg_k$
                 is the first unbroken $b$-gap lying to the immediate right of $ess(ag_i)$ so far;

            } while(Alternate Break($v_c$, $v_c^*$) returns Failure)

        3. Put the maximally complete candidate $v_c^*$ with respect to the kernel $\{v_c\}$ into $S$;

        4. Let $v_c = v_c^*$ and go to Step 2;
```

Proof. By Lemma 3.7, if the procedure Alternate Break returns Success, it means that $\{v_c\}$ and $\{v_c^*\}$ are complete. On the other hand, if the procedure returns Failure, it means that the index of v_c^* is too large to cooperate with v_c; hence, the index of the possible candidate v_c^* gets smaller in each iteration of the do-while loop. In addition, as the forward observation, which propagates from $\{v_c\}$, breaks some b-gaps, the latter remain broken for the possible candidate v_c^* of each subsequent selection. Similarly, the a-gaps broken by the backward observation from the preceding possible kernels $\{v_c^*\}$ will also be broken by $\{v_c^*\}$ for each subsequent selection.

Consider the candidate $v_c^* = v_{c_k}$ selected in Step 2-2, and suppose the forward pointer *a2b(ag_i)* is $bg_{k'}$, where ag_i is the right blocking a-gap of v_c. If $bg_{k'}$ is broken, we need to use the static tree set union and table look-up method in [13] to find the first unbroken b-gap bg_k subsequent to $bg_{k'}$, $k \geq k'$. In addition, assume we select a candidate v_{c_w} other than $v_c^* = v_{c_k}$ and $w > k$. The b-gap bg_k, which is unbroken by the forward observation from the kernel $\{v_c\}$, is a left blocking gap of v_{c_w}, and v_{c_w} cannot break the a-gap ag_i because $ess(ag_i) \leq b_{\ell_k}$. That is, $\{v_c\}$ and $\{v_{c_w}\}$ are incomplete by Corollary 3.8. Therefore, each selection of v_c^* in Algorithm MPDI is a maximally possible candidate with respect to the kernel $\{v_c\}$. This argument is repeated until the procedure Alternate Break returns Success and the maximally complete candidate is found. □

Theorem 3.10 *Given a connected interval graph $G = (V, E)$, Algorithm* MPDI *produces a PDS S of minimum cardinality for G.*

Proof. First, by Lemmas 3.3 and 3.6, v_{c_1} must be in S. Moreover, we know the forward observation from $\{v_c\}$ ($v_c = v_{c_1}$) stops at a-gap ag_i by Lemma 3.3. We have to choose the next candidate v_c^* such that $\{v_c^*\}$ is maximally complete with respect to $\{v_c\}$. By Lemma 3.9, the selection of candidates in Algorithm MPDI finds the maximally complete candidate v_c^* with respect to the kernel $\{v_c\}$ after the procedure Alternate Break returns Success. Then, the candidate v_c^* is included in S, and will play the role of v_c. The process repeats until there is no right blocking a-gap of the last selected candidate v_c^*, since the forward observation from $\{v_c^*\}$ will observe the rest of the vertices in G. The correctness of the algorithm follows. □

Theorem 3.11 *Algorithm* MPDI *takes* $\Theta(n \log n)$ *time, which is asymptotically optimal. In addition, it takes* $O(n)$ *time provided that the given endpoints of the intervals are sorted.*

Proof. We claim that the running time is linear in the total number of a-gaps and b-gaps if the given endpoints of the intervals are sorted. Assume that the total running time is $\mathcal{C} = \mathcal{C}^A + \mathcal{C}^B$, where $\mathcal{C}^A$ and $\mathcal{C}^B$ denote, respectively, the time required to process a-gaps and b-gaps when selecting PMU candidates. Clearly, the running time for a-gaps, $\mathcal{C}^A$, is linear in p, i.e., the number of a-gaps, since the operations that involve a-gaps associated with the forward observations proceed in ascending order without backtracking. Consider the running time $\mathcal{C}^B$ by aggregate analysis. We know that the b-gaps broken by the forward observation in the preceding iterations remain broken in the subsequent iterations. Thus, every b-gap is labeled as "broken" and counted at most once, and the time $\mathcal{C}^B$ is linear in r, i.e., the number of b-gaps. Meanwhile, the forward pointer *a2b*( ) update operations for the associated essential spots of a-gaps can be solved by the method in [13]. The running time is also linear in the total number of a-gaps and b-gaps. Thus, Algorithm MPDI takes linear time if the endpoints of the intervals are sorted. The time bound is due to the sorting of interval endpoints. The algorithm is optimal because we can reduce the *Minimum Gap Problem*, which requires $\Omega(n \log n)$ time under the algebraic computation tree model of Ben-Or [6], to the power domination problem in interval graphs. The reduction scheme is similar to the result reported in [22], and is described as follows.

Minimum Gap Problem: Given $x_1, x_2, \ldots, x_n \in R$ and $\epsilon > 0$, determine if $|x_i - x_j| \geq \epsilon$, for all $i \neq j$.

We map every number x_i into an open interval $(x_i, x_i + \epsilon)$ and obtain an interval representation of an interval graph G. If $\gamma_p(G)$ is n, we answer YES to the original problem; otherwise, we answer NO. This transformation only takes $O(n)$ time; therefore, the reduction builds the $\Omega(n \log n)$ lower bound for the power domination problem in interval graphs. We remark that the reduction scheme also holds for the domination problem in interval graphs. That is, it builds the same lower bound for domination in interval graphs. □

4 Power Dominating Set for Proper Circular-arc Graphs

We refer to the results reported in [20, 22, 35] and consider the power domination problem in circular-arc graphs. A graph G is called a *circular-arc graph* if its vertices can be put into a one-to-one correspondence with a set of arcs on a circle such that two vertices are adjacent in G if and only if their corresponding arcs have nonempty intersections. We call this set of arcs on a circle a *circular-arc representation.* G is a *proper circular-arc graph* if no arc is contained in another arc in G. A circular-arc is denoted by $[a_i, b_i]$, where b_i follows a_i in a clockwise direction, and a_i and b_i are called the left and right endpoints respectively. Note that arc $[b_i, a_i]$ denotes the complement of arc $[a_i, b_i]$ with respect to the circle. We select a right endpoint arbitrarily and label it b_1, and proceed to label the subsequent right endpoints following b_1 in a clockwise direction as $b_2, b_3, \ldots, b_n$; thus, the corresponding vertices have a *circular ordering* $v_1, v_2, \ldots, v_n$ with $b_1 \preceq b_2 \preceq \ldots \preceq b_n \preceq b_1$, where $b_i \preceq b_j$ means that b_j follows b_i in a clockwise direction. Note that $b_i \prec b_j$ means that b_j follows b_i in a clockwise direction and $b_i \neq b_j$. In the following discussion, a circular ordering is given in a clockwise direction, and is omitted if doing so does not cause confusion. Recall that the notions of a-gaps and b-gaps remain the same in a circular-arc graph. It is possible that b_1 may lie in the first b-gap bg_1, i.e., $b_1 \in \{b_{f_1}, \ldots, b_{\ell_1}\}$, and $b_1 \neq b_{f_1}$, where $b_{f_1} \preceq b_n \prec b_1$. In such a case, we re-label b_{f_1} as b_1, the beginning of a new circular ordering, and proceed to re-label the subsequent right endpoints accordingly. Two kernels $\{v_{c_i}\}$ and $\{v_{c_k}\}$, $c_i \prec c_k$, are said to be complete if all the vertices between v_{c_i} and v_{c_k} belong to $\mathcal{G}_{\{v_{c_i}, v_{c_k}\}}$. In addition, $\{v_{c_k}\}$ is said to be maximally complete with respect to $\{v_{c_i}\}$ if they are complete and there exists no other $v_{c_{k'}}$ such that $c_i \prec c_k \prec c_{k'}$, and $\{v_{c_i}\}$ and $\{v_{c_{k'}}\}$ are complete.

All the lemmas in Section 3 also hold for circular-arc graphs. Note that we do not consider the trivial case if an arc intersects all the other arcs; otherwise, we could just select that arc as a PDS and let $\gamma_p(G) = 1$. First, we present a linear time algorithm for the power domination problem in a proper circular-arc graph, where no arc is contained in another arc provided that a circular ordering of the proper circular-arc graph is given. We consider general circular-arc graphs in Section 5. Next, we apply Step 2 of Algorithm MPDI to find for $v_c = v_{c_i}$ the candidate $v_c^* = v_{c_k}$, $c_i \prec c_k$, such that $\{v_c^*\}$ is maximally complete with respect to $\{v_c\}$. Clearly, this process works in circular-arc graphs. We denote this candidate v_c^* as $\mathsf{NEXT}(v_c)$ if $\{v_c^*\}$ is maximally complete with respect to $\{v_c\}$. For the candidate v_c, if no a-gap is the right blocking a-gap of v_c, then all the vertices following v_c clockwise belong to $\mathcal{G}_{\{v_c\}}$ and we let $\mathsf{NEXT}(v_c)$= *null*. We present the following lemma to illustrate the *interleaving property* of the relationship between v_c and $\mathsf{NEXT}(v_c)$.

Lemma 4.1 (Interleaving Property) *Given a circular-arc graph G with a circular ordering $v_1, v_2, \ldots, v_n$, for any two distinct PMU candidates v_{c_i} and v_{c_j}, $c_i \prec c_j$, we have $\mathsf{NEXT}(v_{c_i}) \preceq \mathsf{NEXT}(v_{c_j})$.*

Proof. Assume there exist two distinct PMU candidates v_{c_i} and v_{c_j}, $c_i \prec c_j$, such that $\mathsf{NEXT}(v_{c_j}) \prec \mathsf{NEXT}(v_{c_i})$. Since $c_i \prec c_j$ and $\{v_{c_i}\}$ and $\{\mathsf{NEXT}(v_{c_i})\}$ are complete, $\{v_{c_j}\}$ and $\{\mathsf{NEXT}(v_{c_i})\}$ are complete, which contradicts the fact that $\{\mathsf{NEXT}(v_{c_j})\}$ is maximally complete with respect to $\{v_{c_j}\}$. □

Based on the above key lemma, we use a similar idea to that of Hsu *et al.* [20] and construct a directed graph $D = (V_D, E_D)$, where $V_D = \{v_{c_1}, v_{c_2}, \ldots, v_{c_r}\}$ and a directed edge $\overrightarrow{v_{c_i}v_{c_j}} \in E_D$ if and only if $v_{c_j} = \mathsf{NEXT}(v_{c_i})$, $c_i \prec c_j$. By assumption, G is a connected circular-arc graph with a circular ordering $v_1, \ldots, v_n$. First, we assume that $V_D \neq \emptyset$; that is, there is at least one b-gap in G; otherwise, we let $\{v_n\}$ be a PDS. Next, we assume that every vertex $v_{c_i} \in V_D$ has its $\mathsf{NEXT}(v_{c_i})$; that is, $\mathsf{NEXT}(v_{c_i}) \neq$ *null* for every i; otherwise, if there exists some v_{c_i} with $\mathsf{NEXT}(v_{c_i})$= *null*, we select it as a PDS. Consequently, there exists at least one directed cycle in D because V_D is of finite cardinality. Besides, no two directed cycles can share a common vertex, since every vertex has out-degree exactly one in D. We define $PDS(v_{c_i}) = \{v_{c_i}^{(0)}, v_{c_i}^{(1)}, \ldots, v_{c_i}^{(m-1)}\}$, where $v_{c_i}^{(j+1)}$ =$\mathsf{NEXT}(v_{c_i}^{(j)})$, $v_{c_i}^{(0)} = v_{c_i}$, and $v_{c_i} \preceq \mathsf{NEXT}(v_{c_i}^{(m-1)})$. By definition, $PDS(v_{c_i})$ is a PDS containing v_{c_i} for a circular-arc graph G. We have the following lemma.

Lemma 4.2 *Let S be a PDS of a circular-arc graph G and $v_{c_i} \in S$ for some i. Then, we have $|PDS(v_{c_i})| \leq |S|$.*

Proof. Let the vertices in S and $PDS(v_{c_i})$ be ordered clockwise, and assume that $|PDS(v_{c_i})| > |S|$. There must exist at least an arc $[v_{c_i}^{(j)}, v_{c_i}^{(j+1) \bmod (m)}]$ for some $0 \leq j \leq m-1$, where the arc does not contain any vertex in S. However, this contradicts the definition of NEXT. □

By Lemma 4.2, we know that $PDS(v_{c_i})$ is the minimum PDS containing v_{c_i}. In addition, a vertex v_{c_i} in V_D is called a *valid candidate* if $|PDS(v_{c_i})| = \gamma_p(G)$. The next lemma follows.

Lemma 4.3 *There is at least one directed cycle consisting exclusively of valid candidates in D.*

Proof. By Lemma 4.2, there is a vertex v_{c_i} that is a valid candidate for some c_i; i.e., $|PDS(v_{c_i})| = \gamma_p(G)$. By assumption, v_{c_i} has its own $\mathsf{NEXT}(v_{c_i})$ and $\mathsf{NEXT}(v_{c_i})$ is definitely contained in $PDS(v_{c_i})$. Again, by Lemma 4.2, $\mathsf{NEXT}(v_{c_i})$ is also a valid candidate because $|PDS(\mathsf{NEXT}(v_{c_i}))| \leq |PDS(v_{c_i})| = \gamma_p(G)$.

We repeat the argument until there are two indices, a and b, such that $v_{c_i}^{(a)} = v_{c_i}^{(b)}$, where $a < b$, since each vertex has out-degree exactly one in D and the cardinality of V_D is finite. Thus, there is at least one directed cycle consisting exclusively of valid candidates. □

By the Interleaving Property and the above lemma, we have the following theorem.

Theorem 4.4 *Every directed cycle C in the directed graph D consists exclusively of valid candidates.*

Proof. By Lemma 4.3, we select C^* as the directed cycle consisting exclusively of valid candidates. For a distinct directed cycle C and an arbitrary vertex v_{c_i} in C, we pick the vertex v_{c_k} in C^* to the immediate left of v_{c_i} such that no vertex in $C^* \cup C$ falls in $[v_{c_k}, v_{c_i}]$. Consider $PDS(v_{c_i}) = \{v_{c_i} = v_{c_i}^{(0)}, \ldots, v_{c_i}^{(m-1)}\}$ and $PDS(v_{c_k})$. By the Interleaving Property, we know there must exist a vertex in $PDS(v_{c_k})$ that lies in $[v_{c_i}^{(j)}, v_{c_i}^{(j+1) \bmod (m)}]$ for each j, $0 \leq j \leq m-1$. From the assumption about v_{c_k}, v_{c_k} lies in $[v_{c_i}^{(m-1)}, v_{c_i}^{(0)}]$. Consequently, $|PDS(v_{c_i})| \leq |PDS(v_{c_k})| = \gamma_p(G)$ and v_{c_i} is a valid candidate. Similar to the proof of Lemma 4.3, C consists exclusively of valid candidates. □

Note that the selection of candidate $v_c^* = v_{c_k}$ associated with the b-gap bg_k in Step 2-2 of Algorithm MPDI is exactly the candidate that is maximally complete with respect to $\{v_c\}$ in proper circular-arc graphs. The reason is that if $v_c^* = v_{c_k}$ breaks the a-gap ag_i, it will also break all the a-gaps between ag_i and bg_k because their essential spots follow $ess(ag_i)$ clockwise. The do-while loop condition in Algorithm MPDI, which checks the completeness condition by the procedure Alternate Break, is unnecessary for proper circular-arc graphs. Based on the above discussion, we propose Algorithm MPDPC to solve the power domination problem in proper circular-arc graphs.

Theorem 4.5 *Given a connected proper circular-arc graph $G = (V, E)$, Algorithm* MPDPC *produces a PDS of minimum cardinality for G in linear time if the circular-arc endpoints are sorted.*

Proof. If there are no b-gaps in G, then $\{v_n\}$ is a PDS by Lemma 3.4. Similarly, if there are no right blocking a-gaps of v_c for some PMU candidate v_c, then $\{v_c\}$ is a PDS by Lemma 3.5. The main step, Step 3, which performs the same operation as Step 2 in MPDI to find $\mathsf{NEXT}(v_c)$ for v_c, except for checking the completeness condition by the procedure Alternate Break, also works clockwise in proper circular-arc graphs. As there is a candidate v_c that is visited twice, there exists a directed cycle from v_c to v_c. By Theorem 4.4, v_c is a valid candidate and $PDS(v_c)$ is a PDS of cardinality $\gamma_p(G)$.

For the time complexity analysis, we only need to consider the time cost of Step 3 in Algorithm MPDPC. Since each iteration of finding $\mathsf{NEXT}(v_c)$ for v_c takes constant time and the number of iterations in the while loop is at most $O(n)$, the time complexity is linear in the size of the vertex set of G. □

Algorithm 2: **MPDPC.** Find a minimum PDS of a connected proper circular-arc graph.

Input. A connected proper circular-arc graph $G = (V, E)$ with a circular ordering $v_1, v_2, \ldots, v_n$. A global doubly-linked list $A[\,]$ consisting of all the a-gaps $ag_1, ag_2, \ldots, ag_p$ and their essential spots $ess(ag_i)$, $1 \leq i \leq p$, and a global doubly-linked list $B[\,]$ consisting of all the b-gaps $bg_1, bg_2, \ldots, bg_r$ and their PMU candidates $v_{c_1}, v_{c_2}, \ldots, v_{c_r}$.

Output. A minimum PDS S of G.

Method.

```
1. if(there is no $b$-gap)
        { Let $S = \{v_n\}$ and return $S$; }

2. Let all PMU candidates be labeled "unvisited" and let $v_c = v_{c_1}$;

3. while($v_c$ is "unvisited")
        { Label $v_c$ as "visited" and run Step 2-1 in MPDI clockwise;
          if(there is no right blocking $a$-gap of $v_c$)
             { Let $S = \{v_c\}$ and return $S$; }
          Run Step 2-2 in MPDI clockwise to find $v_c^* =$NEXT$(v_c)$;
          Let $v_c = v_c^*$; }

4. Let $S = PDS(v_c)$ and return $S$;
```

5 Power Dominating Set for Circular-arc Graphs

In this section, we combine the MPDI and MPDPC strategies to extend our result to general circular-arc graphs. To use Algorithm MPDPC in general circular-arc graphs, we must first obtain $\mathsf{NEXT}(v_{c_i})$ for every v_{c_i}, $1 \leq i \leq r$. Since the do-while loop in Step 2 of Algorithm MPDI is necessary for general circular-arc graphs, the time cost of checking the completeness condition in the procedure Alternate Break may not be constant in each iteration in the worst case. Hence, we cannot process each of the PMU candidates $v_{c_1}, v_{c_2}, \ldots, v_{c_r}$ separately.

Based on the Interleaving Property, we partition the PMU candidates $v_{c_1}, v_{c_2}, \ldots, v_{c_r}$ sequentially into q subsets $Q_1, Q_2, \ldots, Q_q$, $1 \leq q \leq r$, such that each subset Q_k consists of v_{c_j}'s that have the same $\mathsf{NEXT}(v_{c_j})$. We denote the candidate $v_{c_j}^*$ as $\mathsf{NEXT}(Q_k)$, $1 \leq k \leq q$ if for every v_{c_j} in Q_k, $v_{c_j}^* = \mathsf{NEXT}(v_{c_j})$. That is, $\{v_{c_j}^*\}$ is maximally complete with respect to $\{v_{c_j}\}$ for every v_{c_j} in Q_k. We have the following properties.

Property 5.1

1. *If two PMU candidates have the same right blocking a-gap, they obviously belong to the same subset Q_k.*

2. *For any two distinct PMU candidates v_{c_i} and v_{c_j}, $c_i \prec c_j$, if their right blocking a-gaps are respectively $ag_{i'}$ and $ag_{j'}$ ($i' \prec j'$) such that $a2b(ag_{i'})$ and $a2b(ag_{j'})$ point respectively to the b-gaps $bg_{i''}$ and $bg_{j''}$ associated with PMU candidates $v_{c_{i''}}$ and $v_{c_{j''}}$ ($c_{j''} \preceq c_{i''}$), then by the Interleaving Property, they belong to the same subset Q_k.*

Based on the above properties, when the forward observation propagates through some PMU candidate v_c, that is, when we process some PMU candidate v_c, the essential spot of the right blocking a-gap ag of v_c determines the possible PMU candidate $\mathsf{NEXT}(v_c)$, and also classifies it into some subset Q_k. More specifically, as the forward observation propagates through v_{c_j} following v_{c_i} clockwise and $ess(ag_{j'})$ lies to the left of $ess(ag_{i'})$ clockwise, we let v_{c_j} belong to the subset Q_k containing v_{c_i} and determine the latest possible PMU candidate $\mathsf{NEXT}(Q_k)$ (=$\mathsf{NEXT}(v_{c_j})$ =$\mathsf{NEXT}(v_{c_i})$) for Q_k. Note that the possible $\mathsf{NEXT}(Q_k)$ is associated with the first unbroken b-gap subsequent to $bg_{j''}$ if the latter is broken by a forward observation. Furthermore, for any v_c preceding v_{c_j} ($c \prec c_j$), if $c_{j''} \preceq c^*$, where v_{c^*} is the latest possible PMU candidate $\mathsf{NEXT}(v_c)$, we merge all the subsets containing any v_c into the subset Q_k containing v_{c_j}.

To obtain $\mathsf{NEXT}(Q_k)$ for every Q_k, we must use the procedure Alternate Break to check the completeness condition. However, in each single iteration i, we avoid executing the entire procedure Alternate Break for v_{c_i} so that the completeness condition is not checked repeatedly. The procedure Alternate Break is divided into several parts and executed in multiple iterations instead. More precisely, consider the alternating break executed by $\{v_{c_i}\}$ and $\{v_{c_{i''}}\}$, where $v_{c_{i''}}$ is associated with b-gap $bg_{i''}$ pointed to by $a2b(ag_{i'})$, and $ag_{i'}$ is the right blocking a-gap of v_{c_i}. The forward observation from v_{c_i} stops at the a-gap $ag_{i'}$, and the backward observation from $v_{c_{i''}}$ stops at the first unbroken b-gap. The observations only propagate alternately once, after which we consider the next PMU candidate $v_{c_{i+1}}$ and repeat this argument. To check the completeness condition between a forward observation and a backward observation, we define a *cross point* for each subset Q_k, denoted by Q^c_k. Consider v_{c_i} in some subset Q_k and its corresponding possible $\mathsf{NEXT}(Q_k)$. After the backward observation from the possible $\{\mathsf{NEXT}(Q_k)\}$ stops at the first unbroken b-gap bg and if bg lies to the immediate left of the a-gap ag_j, we let the cross point $Q^c_k = j$. The cross point Q^c_k of Q_k has the property that if the forward observation from $\{v_{c_i}\}$ for every $v_{c_i} \in Q_k$ propagates and stops at the a-gap in which the cross point Q^c_k lies, then the completeness condition between each PMU candidate v_{c_i} in the subset Q_k and $\mathsf{NEXT}(Q_k)$ holds; that is, $\{\mathsf{NEXT}(Q_k)\}$ is maximally complete with respect to $\{v_{c_i}\}$ for each v_{c_i} in Q_k.

We split the procedure Alternate Break into multiple iterations and modify Algorithm MPDI, as a preprocessing step of Algorithm MPDC to compute $\mathsf{NEXT}(v_{c_i})$ for every v_{c_i}, $1 \leq i \leq r$. Similarly, we label each b-gap bg as "broken" if a forward observation propagates through $ess(bg)$; however, we avoid deleting a-gaps broken by the backward observation from the possible PMU candidate $\mathsf{NEXT}(v_{c_i})$. Note that, in interval graphs, a forward observation propagates through each bg as well as its $ess(bg)$, but it is not a certainty in circular-arc graphs. Since we need to find $\mathsf{NEXT}(v_c)$ for each candidate v_c, i.e., a pair of candidates with the maximally complete property, we only consider the alternating break between v_c and $\mathsf{NEXT}(v_c)$; thus, $ess(bg)$ is considered instead of bg as a forward observation propagates in circular-arc graphs. As a result, we label every b-gap bg as "broken" when a forward observation propagates through $ess(bg)$; and we recover bg "unbroken" in order to find possible candidates in a circular fashion when a forward observation passes through bg. Similarly, we use two doubly-linked lists $A[\ ]$, $B[\ ]$ to store all the a-gaps and b-gaps respectively, and maintain a forward pointer $a2b(ag)$ for each a-gap ag, a back pointer $b2a(bg)$ for each b-gap bg, and pointers to the next a-gap and the previous b-gap for each endpoint (a_i or b_i), as described in Section 3. Without loss of generality, we assume there is no PMU candidate v_c whose $\mathsf{NEXT}(v_c) = null$, i.e., there is no right blocking a-gap for v_c.

Lemma 5.2 *Given a connected circular-arc graph $G = (V, E)$, Algorithm* MPDC-Preprocessing *finds* $\mathsf{NEXT}(v_{c_i})$ *for each PMU candidate* v_{c_i}, $1 \leq i \leq r$.

Proof. Lemma 3.9 shows that Step 2 in Algorithm MPDI, i.e., the iterative procedure Alternate Break, finds $\mathsf{NEXT}(v_c)$ for v_c correctly. We modify this step to consider several PMU candidates together. Steps 2-1 and 2-4 are the same as those in Algorithm MPDI. In Step 2-2, if the forward

Algorithm 3: **MPDC-Preprocessing.** Find $\mathsf{NEXT}(v_c)$ for each candidate v_c in a connected circular-arc graph.

Input. A connected circular-arc graph $G = (V, E)$ with a circular ordering $v_1, v_2, \ldots, v_n$. A global doubly-linked list $A[\ ]$ consisting of all the a-gaps $ag_1, ag_2, \ldots, ag_p$ and their essential spots $ess(ag_i)$, $1 \le i \le p$, and a global doubly-linked list $B[\ ]$ consisting of all the b-gaps $bg_1, bg_2, \ldots, bg_r$ and their PMU candidates $v_{c_1}, v_{c_2}, \ldots, v_{c_r}$. A linked list $Qlink$: $Q_1 \leftarrow Q_2 \leftarrow \ldots \leftarrow Q_r$.

Output. $\mathsf{NEXT}(v_c)$ for each PMU candidate v_c.

Method.

```
1. Let $Q_1, Q_2, \ldots, Q_r$ be empty sets, each of which is associated with cross point $Q_k^c$
   and $\mathsf{NEXT}(Q_k)$, $\forall 1 \le k \le r$, and initialize $m = 0$, where $m = \max\{k \mid Q_k \neq \emptyset\}$;

2. for $i = 1$ to $r$

   2-1. Find the right blocking $a$-gap $ag_{j''}$ of $v_{c_i}$;

   2-2. if($j'' = Q_k^c$ with some $Q_k \neq \emptyset$)
              $\mathsf{NEXT}(Q_k)$ is a feasible candidate for every $Q_k$ with cross point $Q_k^c$;

   2-3. Label each $b$-gap $bg$ as "broken" and recover each $b$-gap $bg^*$ "unbroken" if the
        forward observation from the kernel $\{v_{c_i}\}$ propagates through $ess(bg)$ and $bg^*$
        respectively;

   2-4. Select a possible PMU candidate $v_{c_j}$ associated with the $b$-gap $bg_j$, where $bg_j$
        is the first unbroken $b$-gap lying to the immediate right of $ess(ag_{j''})$ so far;

   2-5. Find the left blocking $b$-gap $bg$ of $v_{c_j}$, which lies to the immediate left of the
        $a$-gap $ag_{j'}$, and let $Q_i^c = j'$;

   2-6. $Q_i = Q_i \cup \{v_{c_i}\}$ and $\mathsf{NEXT}(Q_i) = v_{c_j}$;          /* Create a subset $Q_i$; */
        while($m \ge 1$ and $v_{c_j} \preceq \mathsf{NEXT}(Q_m)$)                   /* Merge the subsets; */
        {     $Q_i = Q_i \cup Q_m$ and let $Q_m = \emptyset$;
              Remove $Q_m$ from $Qlink$ and let $m = m'$, where $Q_{m'} \leftarrow Q_m$ in $Qlink$;
        }
        Let $m = i$;

   End for
```

observation propagates and stops at the a-gap $ag_{Q_k^c}$ with some $Q_k \neq \emptyset$, there will not be any pairs of unbroken a-gaps and b-gaps between $v_c \in Q_k$ and $\mathsf{NEXT}(Q_k)$. The reason is that the b-gaps between the cross point Q_k^c and $\mathsf{NEXT}(Q_k)$ were broken by the preceding forward observation. Therefore, $\{\mathsf{NEXT}(Q_k)\}$ is maximally complete with respect to all the candidates in Q_k, for each Q_k with the same cross point Q_k^c. Step 2-3 is similar except for the case where the forward observation propagates through such bg^*'s. Because we only consider the a-gaps and b-gaps between v_{c_i} and the possible $\mathsf{NEXT}(v_{c_i})$, recovering each b-gap bg^* that lies to the left of $ag_{j''}$ has no influence on the alternating break between v_{c_i} and the possible $\mathsf{NEXT}(v_{c_i})$. We run Step 2-5 to determine the location of the cross point Q_i^c. Step 2-6 is divided into two cases.

Case 1. $\mathsf{NEXT}(Q_m) \prec v_{c_j}$.
If there is an unbroken b-gap between $\mathsf{NEXT}(Q_m)$ and v_{c_j}, then v_{c_i} clearly belongs to a new subset Q_i. On the other hand, suppose the preceding forward observation breaks all the b-gaps between $\mathsf{NEXT}(Q_m)$ and v_{c_j} before we consider v_{c_i}. Then, the preceding forward observation must occur after we process v_{c_m}; otherwise, it would render $\mathsf{NEXT}(Q_m) = v_{c_j}$, which would be a contradiction. Thus, v_{c_i} belongs to a new subset Q_i.

Case 2. $v_{c_j} \preceq \mathsf{NEXT}(Q_m)$.
First, note that the preceding forward observation breaks all the b-gaps between Q_m^c and $\mathsf{NEXT}(Q_m)$. If $v_{c_j} = \mathsf{NEXT}(Q_m)$, then the forward observation breaks all the b-gaps between $ess(ag_{j''})$ and $\mathsf{NEXT}(Q_m)$. Therefore, every $v_c \in Q_m$ belongs to the subset Q_i containing v_{c_i}. On the other hand, the case $v_{c_j} \prec \mathsf{NEXT}(Q_m)$ is equivalent to the condition whereby the procedure Alternate Break returns Failure in Section 3; that is, there is a pair consisting of an unbroken a-gap $ag_{j''}$ and an unbroken b-gap bg_j between v_c and $\mathsf{NEXT}(Q_m)$, where $v_c \in Q_m$. More precisely, the b-gap bg_j lies to the left of Q_m^c. $\mathsf{NEXT}(Q_m)$ is too big to cooperate with $v_c \in Q_m$ and must be replaced by v_{c_j} because of the Interleaving Property. We use the while loop to merge every subset Q_m whose $\mathsf{NEXT}(Q_m)$ is too big, or whose $\mathsf{NEXT}(Q_m)$ is equal to v_{c_j}. Note that the merge-operation can be performed sequentially based on the Interleaving Property. □

Theorem 5.3 *Algorithm* MPDC-Preprocessing *takes linear time if the given circular-arc endpoints are sorted.*

Proof. The proof is similar to that of Theorem 3.11. We claim that the running time is also linear in the total number of a-gaps and b-gaps. Assume that the total running time is $\mathcal{C} = \mathcal{C}^A + \mathcal{C}^B$, where $\mathcal{C}^A$ and $\mathcal{C}^B$ denote, respectively, the time required to process a-gaps and b-gaps when selecting next candidates. Clearly, $\mathcal{C}^A$ is linear in p, i.e., the number of a-gaps, since the operations that involve a-gaps associated with the forward observation proceed in ascending order without backtracking. Consider the running time $\mathcal{C}^B$ in aggregate analysis. Note that the b-gaps broken in the preceding iterations remain broken in the subsequent iterations if the forward observations have not propagated through the right endpoints that define the broken b-gaps. Besides, each b-gap is recovered at most once because the forward observation passes through every arc at most once in a circular fashion. Based on the above discussion, the running time $\mathcal{C}^B$ for labeling b-gaps as "broken" and recovering b-gaps "unbroken" is linear in r, i.e., the number of b-gaps, as we use doubly-linked lists $A[\,]$ and $B[\,]$. Meanwhile, the update operations of the forward pointer *a2b*( ) for the associated essential spots of a-gaps in Step 2-4 can be performed in a similar way by using the method presented in [13]. We can handle the operations for recovering b-gaps "unbroken" via another identical static union tree, since the b-gaps are recovered sequentially in ascending order. Thus, the running time is also linear in the total number of a-gaps and b-gaps.

We also need to consider the operations for subsets $Q_1, Q_2, \ldots, Q_r$ in amortized counting. For each PMU candidate v_c, we insert v_c into an empty subset Q and perform set-union manipulation sequentially in the while loop at most once (i.e., we only count every v_c in the first move). The linked list *Qlink* prevents the while loop from traversing the subsets we have merged already. Hence, Algorithm MPDC-Preprocessing takes linear time if the circular-arc endpoints are sorted. □

After executing Algorithm MPDC-Preprocessing in a given connected circular-arc graph $G = (V, E)$, we can apply Algorithm MPDPC to produce a PDS of minimum cardinality for G in linear time, since the operation for finding $v_c^* = \mathsf{NEXT}(v_c)$ in Step 3 of MPDPC now only takes constant time. The next theorem follows immediately.

Theorem 5.4 *Given a connected circular-arc graph $G = (V, E)$, a PDS of minimum cardinality for G can be obtained by Algorithm* MPDC *(*MPDC-Preprocessing *and* MPDPC*) in linear time if the given circular-arc endpoints are sorted.*

6 Concluding Remarks

We have considered the power domination problem, which is related to the domination problem in graph theory [16], and presented linear time algorithms to solve the power domination problem for both interval graphs and circular-arc graphs, provided that the given endpoints of the corresponding interval representation and circular-arc representation have been sorted. The problem is relevant to many fields. Studying them would be worthwhile because of their applications in real power systems. We conclude the paper with two questions about the power domination problem: What are the complexities of the power domination problem for other classes of intersection graphs? How can the relationship, if any, between the power domination number and other variations of domination numbers be characterized? We will address these questions in our future research.

Acknowledgements. We wish to thank Hengchin Yeh, Ching-Chi Lin, G. J. Chang, and the anonymous reviewers for many valuable comments and suggestions, which helped us improve the quality of the presentation of the paper. The final publication is available at www.springerlink.com.

References

[1] A. Aazami. Domination in graphs with bounded propagation: algorithms, formulations and hardness results, *J. Comb. Optim.* **19(4)** (2010) pp. 429–456.

[2] A. Aazami and M. D. Stilp. Approximation algorithms and hardness for domination with propagation, *SIAM J. Discrete Math.* **23(3)** (2009) pp. 1382–1399.

[3] D. Atkins, T. W. Haynes, and M. A. Henning. Placing monitoring devices in electric power networks modelled by block graphs, *Ars Comb.* **79** (2006) pp. 129–143.

[4] T. L. Baldwin, L. Mili, M. B. Boisen, Jr., and R. Adapa. Power system observability with minimal phasor measurement placement, *IEEE Trans. Power System* **8(2)** (1993) pp. 707–715.

[5] R. Barrera. On the power domination problem in graphs, M.S. Thesis, Texas State University, San Marcos, USA, 2009.

[6] M. Ben-Or. Lower bounds for algebraic computation trees, in *Proc. 15th Annual Symposium on Theory of Computing* (1983) pp. 80–86.

[7] D. J. Brueni. Minimal PMU placement for graph observability: a decomposition approach, M.S. Thesis, Virginia Polytechnic Institute and State University, Blacksburg, USA, 1993.

[8] D. J. Brueni and L. S. Heath. The PMU placement problem, *SIAM J. Discrete Math.* **19(3)** (2005) pp. 744–761.

[9] G. J. Chang. Algorithmic aspects of domination in graphs, in *Handbook of Combinatorial Optimization* (D.-Z. Du and P. M. Pardalos eds.) **3** (1998) pp. 339–405.

[10] P. Dorbec, M. Mollard, S. Klavžar, and S. Špacapan. Power domination in product graphs, *SIAM J. Discrete Math.* **22(2)** (2008) pp. 554–567.

[11] M. Dorfling and M. A. Henning. A note on power domination in grid graphs, *Discrete Applied Math.* **154(6)** (2006) pp. 1023–1027.

[12] R. Emami and A. Abur. Robust measurement design by placing synchronized phasor measurements on network branches, *IEEE Trans. Power Systems* **25(1)** (2010) pp. 38–43.

[13] H. N. Gabow and R. E. Tarjan. A linear-time algorithm for a special case of disjoint set union, *Journal of Computer and System Sciences* **30(2)** (1985) pp. 209–221.

[14] M. C. Golumbic. *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, Inc. (1980).

[15] J. Guo, R. Niedermeier, and D. Raible. Improved algorithms and complexity results for power domination in graphs, *Algorithmica* **52(2)** (2008) pp. 177-202.

[16] T. W. Haynes, S. M. Hedetniemi, S. T. Hedetniemi, and M. A. Henning. Domination in graphs applied to electric power networks, *SIAM J. Discrete Math.* **15(4)** (2002) pp. 519–529.

[17] T. W. Haynes, S. T. Hedetniemi and P. J. Slater. *Domination in Graphs: The Theory*, Marcel Dekker, Inc. New York (1998).

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

[19] W.-K. Hon, C.-S. Liu, S.-L. Peng, and C. Y. Tang. Power domination on block-cactus graphs, in *Proc. the 24th Workshop Combin. Math. and Comput. Theory* (2007) pp. 280–284.

[20] W.-L. Hsu, and K.-H. Tsai. Linear time algorithms on circular-arc graphs, *Inform. Process. Letters* **40(3)** (1991) pp. 123–129.

[21] J. Kneis, D. Mölle, S. Richter, and P. Rossmanith. Parameterized power domination complexity, *Inform. Process. Letters* **98(4)** (2006) pp. 145–149.

[22] D. T. Lee, M. Sarrafzadeh, and Y. F. Wu. Minimum cuts for circular-arc graphs, *SIAM J. Computing* **19(6)** (1990) pp. 1041–1050.

[23] C.-S. Liao and D. T. Lee. Power domination problem in graphs, in *Proc. the 11th International Comput. and Combin. Conference* (2005) pp. 818–828.

[24] K.-P. Lien, C.-W. Liu, C.-S. Yu, and J.-A. Jiang. Transimission network fault location observability with minimal PMU placement, *IEEE Trans. Power Delivery* **21(3)** (2006) pp. 1128–1136.

[25] R. F. Nuqui and A. G. Phadke. Phasor measurement unit placement techniques for complete and incomplete observability, *IEEE Trans. Power Delivery* **20(4)** (2005) pp. 2381–2388.

[26] K.-J. Pai, J.-M. Chang, and Y.-L. Wang. A simple algorithm for solving the power domination problem on grid graphs, in *Proc. the 24th Workshop Combin. Math. and Comput. Theory* (2007) pp. 256–260.

[27] K.-J. Pai, J.-M. Chang, and Y.-L. Wang. Restricted power domination and fault-tolerant power domination on grids, *Discrete Applied Math.* **158(10)** (2010) pp. 1079–1089.

[28] A. G. Phadke. Synchronized phasor measurements in power systems, *IEEE Computer Applications in Power* **6(2)** (1993) pp. 10–15.

[29] J. Peng, Y. Sun, and H. F. Wang. Optimal PMU placement for full network observability using Tabu search algorithm, *Electrical Power and Energy Systems*, **28** (2006) pp. 223–231.

[30] A. G. Phadke, J. S. Thorp, R. F. Nuqui, and M. Zhou. Recent developments in state estimation with phasor measurements, in *Proc. the IEEE Power Systems Conference and Exposition* (2009) pp. 1–7.

[31] D. Raible and H. Fernau. Power domination in $O^*(1.7548^n)$ using reference search trees, in *Proc. the 19th International Symposium Algorithms and Computation* (2008) pp. 136–147.

[32] G. Ramalingam and C. Pandu Rangan. A unified approach to domination problems in interval graphs, *Inform. Process. Letters* **27(5)** (1988) pp. 271–274.

[33] R. E. Tarjan. Efficiency of a good but not linear set union algorithm, *J. ACM* **22(2)** (1975) pp. 215–225.

[34] V. Terzija, G. Valverde, D. Cai, P. Regulski, V. Madani, J. Fitch, S. Skok, M. M. Begovic, and A. G. Phadke. Wide-area monitoring, protection, and control of future electric power networks, in *Proc. the IEEE* **99(1)** (2011) pp. 80–93.

[35] K.-H. Tsai and D. T. Lee. k-Best Cuts for Circular-Arc Graphs, *Algorithmica* **18(2)** (1997) pp. 198–216.

[36] G. Xu, L. Kang, E. Shan, and M. Zhao. Power domination in block graphs, *Theoret. Comput. Science* **359** (2006) pp. 299–305.

[37] M. Zhao, L. Kang, and G. J. Chang. Power domination in graphs, *Discrete Math.* **306(15)** (2006) pp. 1812–1816.