Discrete Applied Mathematics 161 (2013) 1513–1520

Contents lists available at SciVerse ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

Algorithmic aspects of the k-domination problem in graphs☆

James K. Lan [a,*], Gerard Jennhwa Chang [a,b,c]

[a] Department of Mathematics, National Taiwan University, Taipei 10617, Taiwan
[b] Taida Institute for Mathematical Sciences, National Taiwan University, Taipei 10617, Taiwan
[c] National Center for Theoretical Sciences, Taipei Office, Taiwan

ARTICLE INFO

Article history:
Received 21 June 2012
Received in revised form 22 December 2012
Accepted 12 January 2013
Available online 12 February 2013

Keywords:
k-domination
Tree
Block graph
Cactus
Block-cactus graph
Split graph
Algorithm
NP-complete

ABSTRACT

For a positive integer k, a *k-dominating set* of a graph G is a subset $D \subseteq V(G)$ such that every vertex not in D is adjacent to at least k vertices in D. The *k-domination problem* is to determine a minimum k-dominating set of G. This paper studies the k-domination problem in graphs from an algorithmic point of view. In particular, we present a linear-time algorithm for the k-domination problem for graphs in which each block is a clique, a cycle or a complete bipartite graph. This class of graphs includes trees, block graphs, cacti and block-cactus graphs. We also establish NP-completeness of the k-domination problem in split graphs.

1. Introduction

All graphs in this paper are simple, i.e., finite, undirected, loopless and without multiple edges. Domination is a core NP-complete problem in graph theory and combinatorial optimization. It has many applications in the real world such as location problems, sets of representatives, social network theory, etc.; see [3,12] for more interesting applications. A vertex is said to *dominate* itself and all of its neighbors. A *dominating set* of a graph G is a subset D of $V(G)$ such that every vertex not in D is dominated by at least one vertex in D. The *domination number* $\gamma(G)$ of G is the minimum size of a dominating set of G. The *domination problem* is to find a minimum dominating set of a graph.

It is well-known that given any minimum dominating set D of a graph G, one can always remove two edges from G such that D is no longer a dominating set for G [12, p. 184]. The idea of dominating each vertex multiple times is naturally considered. One of such generalizations is the concept of *k-domination*, introduced by Fink and Jacobson in 1985 [10]. For a positive integer k, a *k-dominating set* of a graph G is a subset $D \subseteq V(G)$ such that every vertex not in D is dominated by at least k vertices in D. The *k-domination number* $\gamma_k(G)$ of G is the minimum size of a k-dominating set of G. The *k-domination problem* is to determine a minimum k-dominating set of a graph. The special case when $k = 1$ is the ordinary domination.

Many of the k-domination results in the literature focused on finding bounds on the number $\gamma_k(G)$. In particular, bounds in terms of order, size, minimum degree, maximum degree, domination number, independence number, k-independence number, and matching number were extensively studied [2,4,6,7,9–11,16]; also see the recent survey paper [5].

☆ This research was partially supported by the National Science Council of the Republic of China under grants NSC100-2811-M-002-146 and NSC98-2115-M-002-013-MY3.

* Corresponding author. Tel.: +886 910610006; fax: +886 2 23914439.
E-mail addresses: drjamesblue@gmail.com (J.K. Lan), gjchang@math.ntu.edu.tw (G.J. Chang).

0166-218X/$ – see front matter
doi:10.1016/j.dam.2013.01.015

On the complexity side of the k-domination problem, Jacobson and Peters showed that the k-domination problem is NP-complete for general graphs [14] and gave linear-time algorithms to compute the k-domination number of trees and series–parallel graphs [14]. The k-domination problem remains NP-complete in bipartite graphs or chordal graphs [1]. More complexity results for the k-domination problem are desirable.

In this paper, we explore efficient algorithms for the k-domination problem in graphs. In particular, we present a linear-time algorithm for the k-domination problem in graphs in which each block is a clique, a cycle or a complete bipartite graph. This class of graphs include trees, block graphs, cacti and block-cactus graphs. We also show that the k-domination problem remains NP-complete in split graphs, a subclass of chordal graphs.

2. Preliminaries

Let $G = (V, E)$ be a graph with vertex set V and edge set E. For a vertex v, the *open neighborhood* is the set $N(v) = \{u \in V : uv \in E\}$ and the *closed neighborhood* is $N[v] = N(v) \cup \{v\}$. The *degree* $\deg(v)$ of a vertex v in G is the number of edges incident to v.

The *subgraph of G induced by* $S \subseteq V$ is the graph $G[S]$ with vertex set S and edge set $\{uv \in E : u, v \in S\}$. In a graph $G = (V, E)$, the *deletion of* $S \subseteq V$ *from G*, denoted by $G - S$, is the graph $G[V \setminus S]$. For a vertex v in G, we write $G - v$ for $G - \{v\}$.

In a graph, a *stable set* (or *independent set*) is a set of pairwise nonadjacent vertices, and a *clique* is a set of pairwise adjacent vertices. A *forest* is a graph without cycles. A *tree* is a connected forest. A *leaf* of a graph is a vertex with degree one. A vertex v is a *cut-vertex* if the number of connected components is increased after removing v. A *block* of a graph is a maximal connected subgraph without any cut-vertex. An *end-block* of a graph is a block containing at most one cut-vertex. A *block graph* is a graph whose blocks are cliques. A *cactus* is a connected graph whose blocks are either an edge or a cycle. A *block-cactus graph* is a graph whose blocks are cliques or cycles.

3. The labeling method for *k*-domination

Labeling techniques are widely used in the literature for solving the domination problem and its variants [3,8,13,15]. For k-domination, we employ the following labeling method which is similar to that in [15]. Given a graph G, a *k-dom assignment* is a mapping L that assigns each vertex v in G a two-tuple label $L(v) = (L_1(v), L_2(v))$, where $L_1(v) \in \{B, R\}$, and $L_2(v)$ is a nonnegative integer. Here a vertex v with $L_1(v) = R$ is called a *required vertex*; a vertex v with $L_1(v) = B$ is called a *bound vertex*. An *L-dominating set* of G is a subset $D \subseteq V(G)$ such that

- if $L_1(v) = R$, then $v \in D$, and
- if $L_1(v) = B$, then either $v \in D$ or $|N(v) \cap D| \geq L_2(v)$.

That is, D contains all required vertices, and for each bound vertex v not in D, v is adjacent to at least $L_2(v)$ vertices in D. The *L-domination number* $\gamma_L(G)$ is the minimum size of an L-dominating set in G, such set is called a γ_L-set of G. Notice that if $L(v) = (B, k)$ for all $v \in V(G)$, then $\gamma_L(G) = \gamma_k(G)$. Thus an algorithm for $\gamma_L(G)$ gives $\gamma_k(G)$.

Lemma 1. *Suppose G is a graph with a k-dom assignment* $L = (L_1, L_2)$. *For a vertex* v *in G, let* $G' = G - v$ *and let* L' *be the restriction of L on* $V(G')$ *with the modification that* $L'_2(u) = \max\{L_2(u) - 1, 0\}$ *for* $u \in N(v)$. *If* $L_1(v) = R$ *or* $L_2(v) > \deg(v)$, *then* $\gamma_L(G) = \gamma_{L'}(G') + 1$.

Proof. Suppose D' is a $\gamma_{L'}$-set of G'. Set $D = D' \cup \{v\}$. Since L' is the restriction of L on $V(G')$ with the modification on $L'_2(u)$ and $L_2(u) \leq L'_2(u) + 1$ for $u \in N(v)$, D is clearly an L-dominating set of G. Thus $\gamma_L(G) \leq |D| = |D'| + 1 = \gamma_{L'}(G') + 1$.

Conversely, suppose D is a γ_L-set of G. By the assumption that $L_1(v) = R$ or $L_2(v) > \deg(v)$, v must be included in D. Set $D' = D \setminus \{v\}$. As L' is the restriction of L on G' with the modification on $L'_2(u)$ for $u \in N(v)$, D' is an L'-dominating set of $V(G')$. Hence $\gamma_{L'}(G') + 1 \leq |D'| + 1 = |D| = \gamma_L(G)$. □

This and the following lemma provide an alternative algorithm for the k-domination problem in trees.

Lemma 2. *Suppose G is a graph with a k-dom assignment* $L = (L_1, L_2)$. *For a leaf* v *of G adjacent to u, let* $G' = G - v$ *and let* L' *be the restriction of L on* $V(G')$ *with the modification described below.*

(1) *If* $L(v) = (B, 1)$, *then* $\gamma_L(G) = \gamma_{L'}(G')$, *where* $L'_1(u) = R$.
(2) *If* $L(v) = (B, 0)$, *then* $\gamma_L(G) = \gamma_{L'}(G')$.

Proof. (1) Suppose D' is a $\gamma_{L'}$-set of G'. Since $L'_1(u) = R$, we have $u \in D'$. Then D' is an L-dominating set of G as $|N(v) \cap D'| \geq 1 = L_2(v)$. Thus $\gamma_L(G) \leq |D'| = \gamma_{L'}(G')$.

Conversely, suppose D is a γ_L-set of G. Since $L_2(v) = 1$, either u or v must be included in D. Then clearly $D' = (D \setminus \{v\}) \cup \{u\}$ is an L'-dominating set of G'. Hence $\gamma_{L'}(G') \leq |D'| \leq |D| = \gamma_L(G)$.

(2) Suppose D' is a $\gamma_{L'}$-set of G'. Since L' is the restriction of L on G' and $|N(v) \cap D'| \geq 0 = L_2(v)$, it is clear that D' is an L-dominating set of G. Thus $\gamma_L(G) \leq |D'| = \gamma_{L'}(G')$.

Conversely, suppose D is a γ_L-set of G. If $v \notin D$, then $D' = D$ is an L'-dominating set of G'. If $v \in D$, then $D' = (D \setminus \{v\}) \cup \{u\}$ is an L'-dominating set of G'. Hence $\gamma_{L'}(G') \leq |D'| \leq |D| = \gamma_L(G)$. □

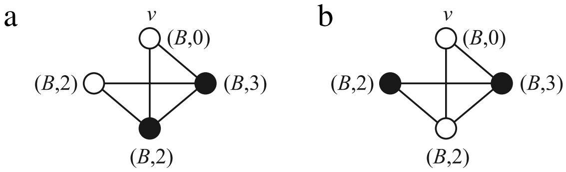

Fig. 1. Minimum L-dominating sets in a graph.

Having these two lemmas at hand, we now establish an alternative algorithm for L-domination in trees. The algorithm will also be used later as a subroutine to find a minimum L-dominating set for cycles, cacti and block-cactus graphs.

Given a tree T of n vertices, it is well-known that T has a vertex ordering $v_1, v_2, \ldots, v_n$ such that v_i is a leaf of $G_i = G[v_i, v_{i+1}, \ldots, v_n]$ for $1 \le i \le n-1$. This ordering can be found in linear-time by using, for example, the breadth-first-search (BFS) algorithm.

The algorithm starts processing a leaf v of a tree T, which is adjacent to a unique vertex u. The label of v is used to possibly relabel u. After v is visited, v is removed from T and a new tree T' is obtained. A linear-time labeling algorithm for finding a γ_L-set in trees is shown as follows.

Algorithm: **kDomTree** (Finding a γ_L-set of a tree)
Input: A tree T of n vertices with a tree ordering $v_1, v_2, \ldots, v_n$ and a k-dom assignment $L = (L_1, L_2)$.
Output: A minimum L-dominating set D of T.
Method:
$D \leftarrow \emptyset$;
for $i \leftarrow 1$ *to* $n-1$ **do**
  let u be the parent of v_i ;
  if $L_1(v_i) = R$ *or* $L_2(v_i) > 1$ **then**
    $L_2(u) \leftarrow \max\{L_2(u) - 1, 0\}$;
    $D \leftarrow D \cup \{v_i\}$;
  if $L(v_i) = (B, 1)$ **then**
    $L_1(u) \leftarrow R$;
end
if $L(v_n) = R$ *or* $L_2(v_n) > 0$ **then**
  $D \leftarrow D \cup \{v_n\}$;

Theorem 3. *Algorithm* **kDomTree** *finds a minimum L-dominating set for a tree in linear-time.*

4. *k*-domination for graphs with special blocks

The main result of this section is an algorithm for the k-domination problem in graphs with tree-like structure. More precisely, we present a linear-time algorithm to find a minimum k-dominating set of a graph whose blocks are cliques, cycles or complete bipartite graphs. These include *block graphs, cacti* and *block-cactus graphs*, etc.

Suppose G is a graph with a k-dom assignment $L = (L_1, L_2)$. For a vertex v of G, denote by $D_v^*(G)$ a minimum L-dominating set of G such that v has the most neighbors in this set, i.e., $\left|N(v) \cap D_v^*(G)\right|$ is maximum. Fig. 1(a) illustrates an example of $D_v^*(G)$, while the minimum L-dominating set formed by the shaded vertices in Fig. 1(b) cannot be selected as $D_v^*(G)$.

Let C be an end-block of G and x be its unique cut-vertex. Denote the end-block C with the modification on $L_1(x) = R$ by C_R. Denote the end-block C with the modification on $L_2(x) = 0$ by C_0. Note that $\gamma_{L'}(C_0) \le \gamma_{L'}(C) \le \gamma_{L'}(C_R) \le \gamma_{L'}(C_0) + 1$, where L' is the restriction of L on $V(C)$ with those modifications.

The construction and correctness of the algorithm is based on the following theorem.

Theorem 4. *Suppose G is a graph with a k-dom assignment $L = (L_1, L_2)$. Let C be an end-block of G and let x be the unique cut-vertex of C. Let G' denote the graph which results from G by deleting all vertices only in C. Let L' and L'' be the restriction of L on G' and C with modifications as described below.*

(1) *If $L_1(x) = R$ or $L_2(x) > \deg_G(x)$, then $\gamma_L(G) = \gamma_{L'}(G') + \gamma_{L''}(C) - 1$.*
(2) *If $L_1(x) = B$ and $\gamma_{L''}(C_0) < \gamma_{L''}(C_R)$, then $\gamma_L(G) = \gamma_{L'}(G') + \gamma_{L''}(C)$, where $L_2'(x) = \max\{L_2(x) - t, 0\}$, $L_2''(x) = t$ and $t = |N(x) \cap D_x^*(C_0)|$.*
(3) *If $L_1(x) = B$ and $\gamma_{L''}(C_0) = \gamma_{L''}(C_R)$, then $\gamma_L(G) = \gamma_{L'}(G') + \gamma_{L''}(C) - 1$, where $L_1'(x) = L_1''(x) = R$.*

Proof. (1) Let D_1 be a $\gamma_{L'}$-set of G' and D_2 be a $\gamma_{L''}$-set of C. Since $L_1'(x) = R$ or $L_2'(x) > \deg_{G'}(x)$, $x \in D_1$; and since $L_1''(x) = R$ or $L_2''(x) > \deg_C(x)$, $x \in D_2$. Clearly $D_1 \cup D_2$ is an L-dominating set of G and we have $\gamma_L(G) \le |D_1 \cup D_2| = \gamma_{L'}(G') + \gamma_{L''}(C) - 1$.

Conversely, suppose D is a γ_L-set of G. By the assumption that $L_1(x) = R$ or $L_2(x) > \deg_G(x)$, we have $x \in D$. It is clear that $D \cap V(G')$ is an L'-dominating set of G' and $D \cap V(C)$ is an L''-dominating set of C. Hence $\gamma_{L'}(G') + \gamma_{L''}(C) - 1 \le \left|D \cap V(G')\right| + |D \cap V(C)| - 1 = |D| = \gamma_L(G)$.

(2) Let D_1 be a $\gamma_{L'}$-set of G' and D_2 be a $\gamma_{L''}$-set of C. If $x \in D_1 \cup D_2$, then clearly $D_1 \cup D_2$ is an L-dominating set of G and hence $\gamma_L(G) \le |D_1 \cup D_2| \le \gamma_{L'}(G') + \gamma_{L''}(C)$. Suppose $x \notin D_1 \cup D_2$. Then

$$\begin{aligned} |N(x) \cap (D_1 \cup D_2)| &= |N(x) \cap D_1| + |N(x) \cap D_2| \\ &\ge L_2'(x) + L_2''(x) \\ &\ge L_2(x) - t + t = L_2(x). \end{aligned}$$

Thus $D_1 \cup D_2$ is an L-dominating set of G and hence $\gamma_L(G) \le \gamma_{L'}(G') + \gamma_{L''}(C)$.

Conversely, suppose D is a γ_L-set of G. We have two cases.

Case 1: $x \in D$. By the assumption that $\gamma_{L''}(C_0) < \gamma_{L''}(C_R)$, we have $|D \cap V(C)| > \gamma_{L''}(C_0)$. Let $D' = (D - V(C)) \cup D_x^*(C_0) \cup \{x\}$. Then $\left|D'\right| = |D|$ and D' is also an L-dominating set of G. Clearly $(D - V(C)) \cup \{x\}$ is an L'-dominating set of G'. Since $\left|N(x) \cap D_x^*(C_0)\right| = t \ge L_2''(x)$, $D_x^*(C_0)$ is an L''-dominating set of C. Thus $\gamma_{L'}(G') + \gamma_{L''}(C) \le |(D - V(C)) \cup \{x\}| + \left|D_x^*(C_0)\right| = \left|D'\right| = |D| = \gamma_L(G)$.

Case 2: $x \notin D$. Then $D \cap V(C)$ must contain an L''-dominating set of C_0. Thus $|D \cap V(C)| \ge \gamma_{L''}(C_0)$. Since D is a γ_L-set of G, we have $|N(x) \cap D| \ge L_2(x)$. Let $D' = (D - V(C)) \cup D_x^*(C_0)$. Then $|D'| = |D|$ and D' is also an L-dominating set of G. Since $|N(x) \cap (D - V(C))| \ge L_2(x) - |D \cap V(C)| \ge L_2(x) - t = L_2'(x)$, it is clear that $D - V(C)$ is an L'-dominating set of G'; since $\left|N(x) \cap D_x^*(C_0)\right| = t \ge L_2''(x)$, it is also that $D_x^*(C_0)$ is an L''-dominating set of C. Therefore $\gamma_{L'}(G') + \gamma_{L''}(C) \le |D - V(C)| + \left|D_x^*(C_0)\right| = \left|D'\right| = |D| = \gamma_L(G)$.

(3) Let $\bar{L}$ be the same as L except for the modification on $\bar{L}_1(x) = R$. We claim that under the assumptions of this case, $\gamma_{\bar{L}}(G) = \gamma_L(G)$. If the claim is true, then by (1), we have the desired result. By definition, clearly $\gamma_{\bar{L}}(G) \ge \gamma_L(G)$. Let D be a γ_L-set of G. By the assumption that $\gamma_{L''}(C_0) = \gamma_{L''}(C_R)$, one can always replace the elements in $D \cap V(C)$ by a $\gamma_{L''}$-set of C_R to get an $\bar{L}$-dominating set of G. Thus $\gamma_{\bar{L}}(G) \le |D| = \gamma_L(G)$. $\square$

We are now ready to present our algorithm, called **kDomG**, to determine a minimum L-dominating set of a graph. Our algorithm takes **kDomB** as a subroutine, which we assume it can find a minimum L'-dominating set of each end-block C of a graph, where L' is the restriction of L on $V(C)$.

Algorithm: **kDomG** (A general approach for finding a γ_L-set in graphs and **kDomB** is a subroutine we assume that can find a $\gamma_{L'}$-set of each end-block of the graph)
Input: A graph G with a k-dom assignment $L = (L_1, L_2)$.
Output: A minimum L-dominating set D of G.
Method:

```
G' ← G;
D ← ∅;
while G' ≠ ∅ do
    if G' is a block then
        D ← D ∪ kDomB(G');
        G' ← ∅;
    else
        let C be an end-block of G' and x be its unique cut-vertex;
        if L_1(x) = R or L_2(x) > deg(x) then
            D ← D ∪ kDomB(C);
        else
            U_0 ← kDomB(C_0);
            U_R ← kDomB(C_R);
            if |U_0| < |U_R| then
                D ← D ∪ D*_x(C_0);
                L_2(x) ← max{L_2(x) − |N(x) ∩ D*_x(C_0)|, 0};
            else // |U_0| = |U_R|
                D ← D ∪ U_R;
                L_1(x) ← R;
        G' ← G' − (V(C) − {x});
end
```

Theorem 5. *Algorithm* **kDomG** *finds a minimum L-dominating set of a graph G in linear-time if* **kDomB** *and $D_x^*(C_0)$ take linear time to compute for each end-block C of G with cut-vertex x.*

Proof. The correctness comes from Theorem 4. For the time complexity, since **kDomG** calls at most three times of **kDomB** and computes D_0^* at most once for each end-block of the graph, it is clear that **kDomG** is linear. □

5. The implementations of the subroutine kDomB in graphs

In this section, we show how to implement the subroutine **kDomB** for some classes of graphs. In particular, we present linear-time algorithms for finding a minimum L-dominating set for complete graphs, cycles and complete bipartite graphs. In addition, the computations of $D_x^*(G)$ for the mentioned classes of graphs are also discussed.

Throughout the rest of this section, suppose $G = (V, E)$ is a graph with a k-dom assignment $L = (L_1, L_2)$. Define

$$\widetilde{R} = \{v \in V : L_1(v) = R \text{ or } L_2(v) > \deg_G(v)\} \text{ and let } |\widetilde{R}| = r.$$

Let D be a minimum L-dominating set of G. By the definition of L-dominating set, all vertices of $\widetilde{R}$ must be included in D.

5.1. Complete graphs

Assume G is a complete graph. Suppose $|V| = n$ and $|\widetilde{R}| = r$. Let $V \setminus \widetilde{R} = \{u_1, u_2, \ldots, u_{n-r}\}$. If $u_i \notin D$ and $u_j \in D$ with $L_2(u_i) > L_2(u_j)$ for some $u_i, u_j \in V \setminus \widetilde{R}$, then $(D - u_j) \cup u_i$ is also a minimum L-dominating set of G. This indicates that we shall choose vertices in $V \setminus \widetilde{R}$ with L_2 value as large as possible. Let t be the minimum index in $\{0, 1, \ldots, n - r\}$ such that $t \geq L_2(u_{t+1}) - r$. It is the case that $D = \widetilde{R} \cup \{u_i \in V \setminus \widetilde{R} : 1 \leq i \leq t\}$.

Algorithm: **kDomKn** (Finding a γ_L-set of a complete graph)
Input: A complete graph $G = (V, E)$ with a k-dom assignment $L = (L_1, L_2)$.
Output: A minimum L-dominating set D of G.
Method:
$D \leftarrow \emptyset$;
let $u_1, u_2, \ldots, u_{n-r}$ be a vertex ordering of $V \setminus \widetilde{R}$ such that $L_2(u_1) \geq L_2(u_2) \geq \ldots \geq L_2(u_{n-r})$;
$u_0 \leftarrow \emptyset$;
$L_2(u_{n-r+1}) \leftarrow r$;
for $t = 0$ *to* $n - r$ **do**
  if $t \geq L_2(u_{t+1}) - r$ **then break**;
end
$D \leftarrow \widetilde{R} \cup \{u_i \in V \setminus \widetilde{R} : 0 \leq i \leq t\}$;

Theorem 6. *Algorithm* **kDomKn** *finds a minimum L-dominating set for a complete graph in linear time.*

Proof. The correctness is clear and is omitted. The time complexity is bound by the computation of the vertex ordering of $V \setminus \widetilde{R}$. Note that $L_2(v) \leq \deg_G(v) < n$ for all $v \in V \setminus \widetilde{R}$. Since each $L_2(v)$ is an integer in the range 0 to n and there are at most n integers need to sort, one can use linear-time sorting algorithms, for examples, Counting sort or Bucket sort, to obtain the vertex ordering. □

Consider the computation of $D_x^*(G)$ of a complete graph G for some fixed vertex x. Since each pair of vertices of G is adjacent, any minimum L-dominating set D of G has the property that $|N(x) \cap D|$ is maximum, and can be selected as $D_x^*(G)$. Thus $D_x^*(G)$ can be found in linear time.

5.2. Cycles

Assume G is a cycle. We will use **kDomTree**, introduced in Section 3, as a subroutine to find a minimum L-dominating set of G. First consider the case of $\widetilde{R} \neq \emptyset$. Let v be a vertex in $\widetilde{R}$. Since v must be included in D, the L_2 values of the neighbors of v should be decreased by 1. Let P_v be the path of $C - v$ with the mentioned modifications on the L_2 values of the neighbors of v. By Lemma 1, the union of any minimum L'-dominating set of P_v and v forms a minimum L-dominating set of G.

Now assume $\widetilde{R} = \emptyset$. If $L_2(v) = 0$ for all $v \in V$, then clearly $D = \emptyset$. Otherwise suppose $L_2(v) > 0$ for some $v \in V$. Let u, w be the two neighbors of v on C. In this case, either v is in D or at least one of u and w is in D. Thus, $|D| = \min\{|\mathbf{kDomTree}(P_v)|, |\mathbf{kDomTree}(P_u)|, |\mathbf{kDomTree}(P_w)|\} + 1$.

The time complexity is clearly linear, since **kDomTree** is linear and it calls at most three times of **kDomTree**.

Now consider the computation of $D_x^*(G)$ of a cycle G for some fixed vertex x. By the definition of $D_x^*(G)$, $\left|N(x) \cap D_x^*(G)\right| \leq 2$. Let y and z be the neighbors of x in G. If $\left|N(x) \cap D_x^*(G)\right| = 2$, then $y, z \in D_x^*(G)$ and $|D_x^*(G)| = \gamma_{L'}(G)$, where L' is the same as L with the modifications on $L_1(y) = R$ and $L_1(z) = R$. If $\left|N(x) \cap D_x^*(G)\right| = 1$ and suppose $y \in D_x^*(G)$, then $|D_x^*(G)| = \gamma_{L'}(G)$, where L' is the same as L with the modifications on $L_1(y) = R$. Thus one can find $D_x^*(G)$ by examining among all possible combinations of modifications on $L'_1(v) = R$, $v \in N[x]$, with $\gamma_{L'}(G) = \gamma_L(G)$. The computation of $D_x^*(G)$ clearly can be done in linear time.

Algorithm: **kDomCYC** (Finding a γ_L-set of a cycle)
Input: A cycle $G = (V, E)$ with a k-dom assignment $L = (L_1, L_2)$.
Output: A minimum L-dominating set D of G.
Method:
$D \leftarrow \emptyset$;
if $\widetilde{R} \neq \emptyset$ **then**
  choose $v \in \widetilde{R}$;
  let P_v be the path of $G - v$ with the modifications on $L_2(u) \leftarrow \max\{L_2(u) - 1, 0\}$ for each $u \in N(v)$;
  $D \leftarrow$ **kDomTree**$(P_v) \cup \{v\}$;
else
  if *there is a vertex v such that $L_2(v) > 0$* **then**
    foreach $u \in N[v]$ **do**
      $D_u \leftarrow$ **kDomCYC**(G_u^R), where G_u^R is the same as G with the modification on $L_1(u) = R$;
    end
    $D \leftarrow D_u$ with minimum $|D_u|$, $u \in N[v]$;
  end

Theorem 7. *Algorithm* **kDomCYC** *finds a minimum L-dominating set in a cycle in linear time.*

5.3. Complete bipartite graphs

Now consider a complete bipartite graph G whose vertex set is a disjoint union of two independent sets A and B. Let $r_1 = |A \cap \widetilde{R}|$, $r_2 = |B \cap \widetilde{R}|$, $A \setminus \widetilde{R} = \{a_1, a_2, \ldots, a_{|A|-r_1}\}$ and $B \setminus \widetilde{R} = \{b_1, b_2, \ldots, b_{|B|-r_2}\}$. If r_1 is no less than $L_2(b)$ for all $b \in B \setminus \widetilde{R}$ and r_2 is no less than $L_2(a)$ for all $a \in A \setminus \widetilde{R}$, then each vertex v not in $\widetilde{R}$ has at least $L_2(v)$ neighbors in $\widetilde{R}$. Otherwise, D must contain some vertices in $(A \cup B) \setminus \widetilde{R}$. Again, if D contains some a_i (resp. b_i), then it is better to choose a_i (resp. b_i) with $L_2(a_i)$ (resp. $L_2(b_i)$) as large as possible. Suppose D contains exactly i vertices in $A \setminus \widetilde{R}$, where $0 \leq i \leq |A| - r_1$. Then D must contains at least j vertices in $B \setminus \widetilde{R}$ and $L_2(b_{j+1}) - r_1 \leq i$, where $j = L_2(a_{i+1}) - r_2$. In addition, we can assume $D \cap (A \setminus \widetilde{R}) = \{a_0, \ldots, a_i\}$, where $a_0 = \emptyset$. Since the algorithm examines all possible choices of i, D is clearly a minimum L-dominating set of G. See Fig. 2 for illustrations.

Algorithm: **kDomKmn** (Finding a γ_L-set of a complete bipartite graph)
Input: A complete bipartite graph G whose vertex set is a disjoint union of two independent sets A and B, and a k-dom assignment $L = (L_1, L_2)$.
Output: A minimum L-dominating set D of G.
Method:
$D \leftarrow \emptyset$;
if $r_1 \geq \max\{L_2(b) : b \in B \setminus \widetilde{R}\}$ *and* $r_2 \geq \max\{L_2(a) : a \in A \setminus \widetilde{R}\}$ **then**
  $D \leftarrow \widetilde{R}$;
else
  $r_1 \leftarrow |A \cap \widetilde{R}|$; $r_2 \leftarrow |B \cap \widetilde{R}|$;
  let $a_1, a_2, \ldots, a_{|A|-r_1}$ and $b_1, b_2, \ldots, b_{|B|-r_2}$ be vertex orderings of $A \setminus \widetilde{R}$ and $B \setminus \widetilde{R}$, respectively, such that $L_2(a_1) \geq L_2(a_2) \geq \ldots \geq L_2(a_{|A|-r_1})$ and $L_2(b_1) \geq L_2(b_2) \geq \ldots \geq L_2(b_{|B|-r_2})$;
  $a_0 \leftarrow \emptyset$; $b_0 \leftarrow \emptyset$; $L_2(a_{|A|-r_1+1}) \leftarrow r_2$; $L_2(b_{|B|-r_2+1}) \leftarrow r_1$;
  $size \leftarrow \infty$;
  for $i = 0$ *to* $|A| - r_1$ **do**
    $j \leftarrow L_2(a_{i+1}) - r_2$;
    if $i + j < size$ *and* $L_2(b_{j+1}) - r_1 \leq i$ **then**
      $t \leftarrow i$;
      $size \leftarrow i + j$;
    end
  end
$D \leftarrow \widetilde{R} \cup \{a_i \in A \setminus \widetilde{R} : 0 \leq i \leq t\} \cup \{b_i \in B \setminus \widetilde{R} : 0 \leq i \leq L_2(a_{t+1}) - r_2\}$;

Theorem 8. *Algorithm* **kDomKmn** *finds a minimum L-dominating set in a complete bipartite graph in linear time.*

Proof. The correctness is clear and is omitted. The time complexity is linear, since the vertex orderings of $A \setminus \widetilde{R}$ and $B \setminus \widetilde{R}$ can be found by using linear-time sorting algorithms, and it takes $O(|V|)$ time on the for-loop of the algorithm. □

Now consider how to compute $D_x^*(G)$ of a complete bipartite graph G for some fixed vertex x. W.L.O.G., suppose $x \in A$. First run **kDomKmn** to get the size, say d, of a γ_L-set of G. Then $D_x^*(G)$ can be found by checking the validity of

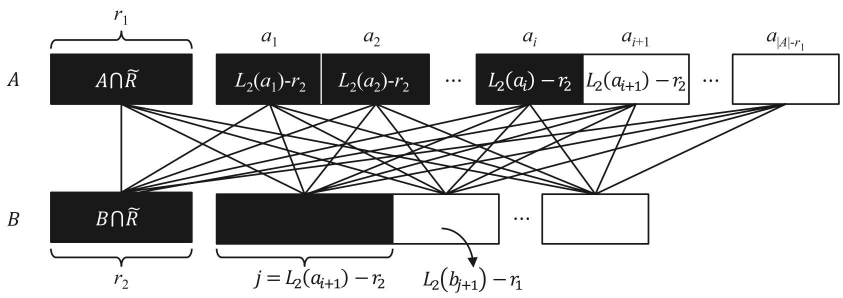

Fig. 2. Finding a minimum L-dominating set in a complete bipartite graph.

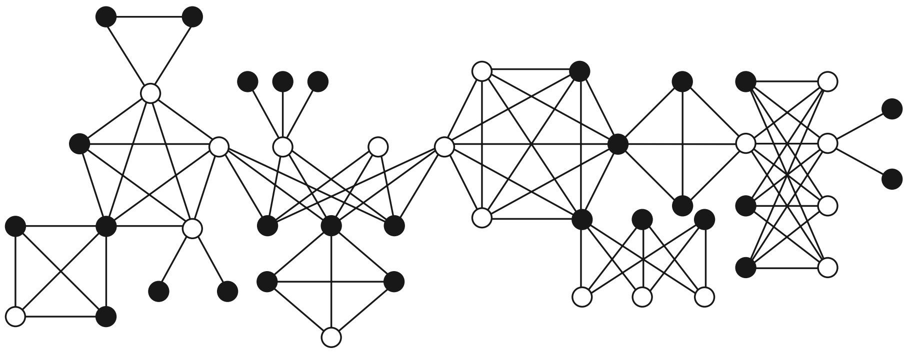

Fig. 3. k-domination in a graph with special blocks; $k = 3$.

$\widetilde{R} \cup \{a_0, \ldots, a_i\} \cup \{b_0, \ldots, b_{d-r-i}\}$ for all $0 \le i \le d - r$, and picking the set with maximum $d - r - i$. The process clearly can be done in linear time.

It is well-known that block graphs, cacti, block-cactus graphs can be recognized in linear-time. By Theorems 5–8, one can immediately have the following result.

Theorem 9. *Algorithm* **kDomG** *finds a minimum L-dominating set in linear time for graphs in which each block is a clique, a cycle or a complete bipartite graph, including block graphs, cacti and block-cactus graphs.*

Fig. 3 shows an example of k-domination in a graph in which the graph contains complete graphs, cycles, and complete bipartite graphs as blocks.

6. NP-completeness results

In this section, we study the complexity of the k-domination problem:

k-DOMINATION

INSTANCE:A graph $G = (V, E)$ and positive integers k and s.

QUESTION: Does G have a k-dominating set of size $\le s$?

It has been proved that the k-domination is NP-complete for general graphs [14], bipartite graphs [1] and chordal graphs [1], in which the reductions are mainly from domination problem for the same class of graphs. In this section, we show that the k-domination remains NP-complete for split graphs, a subclass of chordal graphs. A *split graph* is a graph whose vertex set is the disjoint union of a clique and a stable set. Our reduction is from a well-known NP-complete problem, *vertex cover problem* for general graphs. A *vertex cover* of a graph $G = (V, E)$ is a subset $C \subseteq V$ such that for every edge $uv \in E$ we have $u \in C$ or $v \in C$. The vertex cover problem is to find a minimum vertex cover of G.

VERTEX-COVER

INSTANCE: A graph $G = (V, E)$ and a nonnegative integer c.

QUESTION: Does G have a vertex cover of size $\le c$?

Theorem 10. *For any fixed positive integer* k, k-DOMINATION *is NP-complete for split graphs.*

Proof. Obviously k-DOMINATION belongs to NP, since it is easy to verify a "yes" instance of k-DOMINATION in polynomial time. The reduction is from the vertex cover problem. Let $G = (V, E)$ be an instance of VERTEX-COVER. We construct the

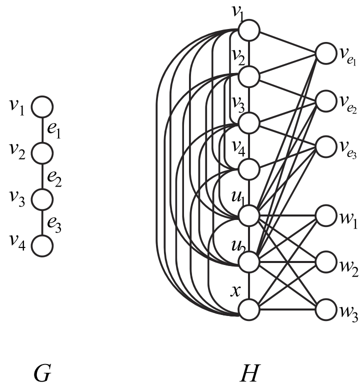

Fig. 4. A transformation to a split graph when $k = 3$.

graph $H = (V_H, E_H)$ with vertex set $V_H = V \cup V_E \cup U \cup W \cup \{x\}$, where $V_E = \{v_e : e \in E\}$, $U = \{u_1, u_2, \ldots, u_{k-1}\}$, $W = \{w_1, w_2, \ldots, u_k\}$, and edge set

$$\begin{aligned} E_H = {} & \{vv_e : v \in V, v_e \in V_E, v \in e\} \\ & \cup \ \{u_i v_e : u_i \in U, v_e \in V_E\} \\ & \cup \ \{uv : u \in U \cup \{x\}, v \in W\} \\ & \cup \ \{uv : u, v \in V \cup U \cup \{x\}, u \neq v\}. \end{aligned}$$

Clearly, H is a split graph whose vertex set is a disjoint union of a clique $V \cup U \cup \{x\}$ and a stable set $V_E \cup W$, and H can be constructed in polynomial time. Now we shall show that G has a vertex cover of size c if and only if H has a k-dominating set of size $c + k$.

Suppose G has a vertex cover C of size c. Then choose $D_H = C \cup U \cup \{x\}$. It is not hard to verify that D_H is a k-dominating set of H of size $c + k$.

On the other hand, suppose D_H is a k-dominating set of H of size $c + k$. If $w_j \notin D_H$ for some $w_j \in W$, then D_H must contain x and all vertices of U. If D_H contains all vertices of W, then $(D_H \setminus W) \cup U \cup \{x\}$ is also a k-dominating set. Thus, we may assume D_H contains x and all vertices of U. Set $D = D_H \cap (V \cup V_E)$. We have $|D| \leq c$. Suppose D contains some $v_e \in V_E$. Choose (arbitrarily) a vertex $v \in e$. Then $(D - v_e) \cup \{v\}$ is also a k-dominating set of H. Thus, we may assume $D \cap V_E = \emptyset$. As a result, adding enough vertices to D results in G a vertex cover of size c (see Fig. 4). □

Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper.

References

[1] T.J. Bean, M.A. Henning, H.C. Swart, On the integrity of distance domination in graphs, The Australasian Journal of Combinatorics 10 (1994) 29–43.
[2] Y. Caro, Y. Roditty, A note on the k-domination number of a graph, International Journal of Mathematics and Mathematical Sciences 13 (1990) 205–206.
[3] G.J. Chang, Algorithmic aspects of domination in graphs, in: D. Du, P.M. Pardalos (Eds.), Handbook of Combinatorial Optimization, 1998, pp. 339–405.
[4] M. Chellali, O. Favaron, A. Hansberg, L. Volkmann, On the p-domination, the total domination and the connected domination numbers of graphs, Journal of Combinatorial Mathematics and Combinatorial Computing 73 (2010) 65–75.
[5] M. Chellali, O. Favaron, A. Hansberg, L. Volkmann, k-domination and k-independence in graphs: a survey, Graphs and Combinatorics 28 (2012) 1–55.
[6] B. Chen, S. Zhou, Upper bounds for f-domination number of graphs, Discrete Mathematics 185 (1998) 239–243.
[7] E.J. Cockayne, B. Gamble, B. Shepherd, An upper bound for the k-domination number of a graph, Journal of Graph Theory 9 (1985) 533–534.
[8] E.J. Cockayne, S.E. Goodman, S.T. Hedetniemi, A linear algorithm for the domination number of a tree, Information Processing Letters 4 (1975) 41–44.
[9] O. Favaron, A. Hansberg, L. Volkmann, On k-domination and minimum degree in graphs, Journal of Graph Theory 57 (2008) 33–40.
[10] J.F. Fink, M.S. Jacobson, Graph Theory with Applications to Algorithms and Computer Science, John Wiley & Sons, Inc., New York, NY, USA, 1985, pp. 283–300.
[11] A. Hansberg, L. Volkmann, Upper bounds on the k-domination number and the k-Roman domination number, Discrete Applied Mathematics 157 (2009) 1634–1639.
[12] T. Haynes, S. Hedetniemi, P. Slater, Fundamentals of Domination in Graphs, in: Monographs and Textbooks in Pure and Applied Mathematics, Marcel Dekker, 1998.
[13] S. Hedetniemi, R. Laskar, J. Pfaff, A linear algorithm for finding a minimum dominating set in a cactus, Discrete Applied Mathematics 13 (1986) 287–292.
[14] M.S. Jacobson, K. Peters, Complexity questions for n-domination and related parameters, in: Eighteenth Manitoba Conference on Numerical Mathematics and Computing (Winnipeg, MB, 1988), Congressus Numerantium 68 (1989) 7–22.
[15] C.S. Liao, G.J. Chang, Algorithmic aspect of k-tuple domination in graphs, Taiwanese Journal of Mathematics 6 (2003) 415–420.
[16] C. Stracke, L. Volkmann, A new domination conception, Journal of Graph Theory 17 (1993) 315–323.