Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in DISCRETE MATHEMATICS

Electronic Notes in Discrete Mathematics 62 (2017) 249–254 www.elsevier.com/locate/endm

A linear-time algorithm for the identifying code problem on block graphs [1]

Gabriela R. Argiroffo [a,2] Silvia M. Bianchi [a,3]
Yanina Lucarini [a,b,4] Annegret K. Wagler [c,5]

[a] *Dept. de Matemática, Universidad Nacional de Rosario, Rosario, Argentina*
[b] *CONICET, Argentina*
[c] *LIMOS, University Clermont Auvergne, Clermont-Ferrand, France*

Abstract

The identifying code problem is a special search problem, challenging both from a theoretical and from a computational point of view, even for several graphs where other usually hard problems are easy to solve, like bipartite graphs or chordal graphs. Hence, a typical line of attack for this problem is to determine minimum identifying codes of special graphs. In this work we study the problem of determining the cardinality of a minimum identifying code in block graphs (that are diamond-free chordal graphs). We present a linear-time algorithm for this problem, as a generalization of a linear-time algorithm proposed by Auger in 2010 for the case of trees. Thereby, we provide a subclass of chordal graphs for which the identifying code problem can be solved in linear time.

Keywords: identifying codes, block graphs, computational complexity

[1] This work was partially supported by PICT 2013-0586 ANPCyT Argentina.
[2] Email: garua@fceia.unr.edu.ar
[3] Email: sbianchi@fceia.unr.edu.ar
[4] Email: lucarini@fceia.unr.edu.ar
[5] Email: annegret.wagler@uca.fr

https://doi.org/10.1016/j.endm.2017.10.043

1 Introduction

Many search problems as, e.g., fault detection in networks or fire detection in buildings, can be modeled by so-called identifying codes in graphs [10].

Consider a graph $G = (V, E)$ and denote by $N[i] = \{i\} \cup N(i)$ the closed neighborhood of a vertex i. A subset $C \subseteq V$ is *dominating* (resp. *identifying*) if $N[i] \cap C$ are non-empty (resp. distinct) sets for all $i \in V$. An *identifying code* of G is a vertex subset which is dominating and identifying.

Not every graph G admits an identifying code, i.e. is *identifiable*: this holds if and only if there are no true twins in G, i.e., there is no pair of distinct vertices $i, j \in V$ with $N[i] = N[j]$ [10]. On the other hand, the whole vertex set of every identifiable graph trivially forms an identifying code.

The *identifying code number* $\gamma_{ID}(G)$ of a graph G is the minimum cardinality of an identifying code of G. Determining $\gamma_{ID}(G)$ is in general NP-hard [6] and remains hard for several graph classes where other in general hard problems are easy to solve, including bipartite graphs [6] and two classes of chordal graphs, namely split graphs and interval graphs [7].

The identifying code problem has been actively studied during the last decade, where typical lines of attack are to determine minimum identifying codes of special graphs or to provide bounds on their size. Closed formulas for the exact value of $\gamma_{ID}(G)$ have been found so far only for restricted graph families (e.g. for paths and cycles [5], for stars [8], for complete multipartite graphs [1] and some subclasses of split graphs [2]).

A linear-time algorithm to determine $\gamma_{ID}(G)$ if G is a tree was provided by Auger [3]. In this paper, we determine the identifying code number of block graphs (that are diamond-free chordal graphs [4]). We present a linear-time algorithm for this problem, as a generalization of the linear-time algorithm by Auger for trees. Thereby, we provide a subclass of chordal graphs for which the identifying code problem can be solved in linear time.

A *block graph* is a graph in which every maximal 2-connected subgraph (block) is a clique (see Fig. 1). Block graphs are precisely those chordal graphs in which every two maximal cliques have at most one vertex in common [9]. Note that a block graph B is identifiable (i.e. has no true twins) if and only if each maximal clique K of B satisfies that all vertices in K, except at most one, have a neighbor that is not in $V(K)$. Moreover, if we call $V(K')$ such neighbors then C is an identifying code of B if and only if $V(K') \subset C$.

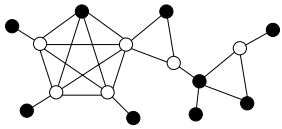

Fig. 1. A block graph B (the black vertices form an identifying code of B).

2 The algorithm

In order to provide a linear-time algorithm which computes $\gamma_{ID}(B)$ of an identifiable block graph B, we adopt the following notation from [3]. Let $G = (V, E)$ be a graph and $v \in V$. Then $C \subseteq V$ is a $\{v\}$-*almost identifying code of* G if the sets $C \cap N[u]$ are nonempty and pairwise distinct for all $u \in V - \{v\}$. Moreover, we say that C satisfies the property

- ID (for identifying) if C is an identifying code in G,
- CO (for code) if $v \in C$,
- ADJ (for adjacent) if v has a neighbour in C,
- FN (for favoured neighbour) if v has a neighbour w with $N[w] \cap C = \{v\}$,
- $\overline{P}$ if C does not satisfy property $P \in \{ID, CO, ADJ, FN\}$.

Let us call P_i any of the above properties and denote by $\gamma_{P_1,\ldots,P_k}(v, G)$ the function that returns the minimum size of a $\{v\}$-almost identifying code in G satisfying P_i with $i = 1, \ldots, k$ or ∞ if no such code exists. We are particularly interested in the ten functions listed in Table 1 since it can be shown that

$$\gamma_{ID}(v, G) = \min(f_1(v, G), f_2(v, G), f_3(v, G), f_4(v, G)). \tag{1}$$

Our main algorithm ICB randomly selects a vertex v_1 from a connected block graph B and calls RICB(v_1, B) that computes the values of all ten functions in a recursive manner in smaller and smaller block graphs.

Algorithm ICB

Input: a connected block graph B and its list of maximal cliques.

Output: $\gamma_{ID}(B)$.

1: randomly select a vertex v_1 and call RICB(v_1, B);

2: return $\gamma_{ID}(v_1, B) = \min(f_1(v_1, B), f_2(v_1, B), f_3(v_1, B), f_4(v_1, B))$.

Algorithm RICB chooses a maximal clique K with vertices $\{v_1, \ldots, v_k\}$ and either returns the initial function values $f_j(v_1, \{v_1\})$ if $K = \{v_1\}$ or deletes all edges of K and calls recursively RICB(v_i, B_i) for all so-obtained components B_i of $B - E(K)$ to compute the list L_i for (v_i, B_i) with $i \in \{1, \ldots, k\}$.

Name	Function	$f_j(v,\{v\})$	Name	Function	$f_j(v,\{v\})$
f_1	$\gamma_{ID,CO,ADJ,FN}$	∞	f_6	$\gamma_{CO,ADJ,\overline{FN}}$	∞
f_2	$\gamma_{ID,CO,ADJ,\overline{FN}}$	∞	f_7	$\gamma_{CO,\overline{ADJ},FN}$	∞
f_3	$\gamma_{ID,CO,\overline{ADJ}}$	1	f_8	$\gamma_{CO,\overline{ADJ},\overline{FN}}$	1
f_4	$\gamma_{ID,\overline{CO},ADJ}$	∞	f_9	$\gamma_{\overline{CO},ADJ}$	∞
f_5	$\gamma_{CO,ADJ,FN}$	∞	f_{10}	$\gamma_{\overline{CO},\overline{ADJ}}$	0

Table 1
List L of functions f_j together with their initial values $f_j(v,\{v\})$.

Algorithm RICB

Input: a block graph B, its list of maximal cliques and $v_1 \in V(B)$.
Output: the list L of the values of the ten functions f_j on (v_1, B).

1: **if** v_1 has degree 0 in B **then**
2: initialize L (Table 1);
3: **else**
4: let K be a maximal clique with $V(K) = \{v_1, \ldots, v_k\}$ and delete its edges;
5: let $B_1, \ldots, B_k$ be the remaining block graphs, resp., containing $v_1, \ldots, v_k$;
6: let $L_i = RICB(v_i, B_i)$ for all $i \in \{1, \ldots, k\}$;
7: compute the ten functions on (v_1, B) from L_i for all $i \in \{1, \ldots, k\}$ (Theorem 2.1).
8: **end if**
9: return the list L of the values of the ten functions f_j on (v_1, B).

We can show:

Theorem 2.1 *For each of the ten functions f_j, we can compute $f_j(v_1, B)$ from $L_i(v_i, B_i)$ for all $i \in \{1, \ldots, k\}$ in time $O(k)$.*

Algorithm RICB returns the values $f_j(v_1, B)$ for $j \in \{1, \ldots, 10\}$ for the original block graph B to ICB, so that the identifying code number of B is obtained by computing the minimum among $f_1(v_1, B)$, $f_2(v_1, B)$, $f_3(v_1, B)$ and $f_4(v_1, B)$ or ∞ if no such code exists.

Theorem 2.2 *Algorithm ICB computes in linear time $\gamma_{ID}(B)$ for an identifiable block graph B (or returns ∞ if no identifying code exists in B).*

In fact, ICB has linear running time $O(n)$ with $n = |V(B)|$. While executing ICB, the number n_Q of maximal cliques of B determines the number of decomposition steps (lines 4 and 5) and recomposition steps (line 7) that are performed during the recursion. In addition, the size k of the maximal cliques

K determines the effort $O(k)$ for the decomposition step and to calculate the function values f_j for the recomposition step. This leads to $2 \cdot O(n_Q) \cdot O(k)$ for the complexity of ICB.

In order to express $O(n_Q)$ and $O(k)$ as functions of n, we note the following. On the one hand, a block graph B has at most linearly many maximal cliques (since two maximal cliques have at most one vertex in common by [9]), i.e., $n-1$ such cliques if B is a tree. Thus, n_Q is at most of order $O(n)$. On the other hand, the size $\omega(B)$ of a maximum clique equals n if B is a clique. Thus, $\omega(B)$ is at most of order $O(n)$, too.

However, it is clear that the two values n_Q and $\omega(B)$ can never attain their maximum value at the same time:

- If n_Q attains its maximum value and thus it is of order $O(n)$, then $\omega(B)$ is necessarily small (in the extreme case, B is a tree and we have $n_Q = n-1$ and $\omega(B) = 2$); hence there is a small fixed value p with $\omega(B) \leq p$, which leads to $2 \cdot O(n) \cdot O(1) = O(n)$.
- If $\omega(B)$ attains its maximum value and thus it is of order $O(n)$, then n_Q is necessarily small (in the extreme case, B is a clique and we have $\omega(B) = n$ and $n_Q = 1$); hence there is a small fixed value p with $n_Q \leq p$, which leads to $2 \cdot O(1) \cdot O(n) = O(n)$.
- If both n_Q and $\omega(B)$ have intermediate values, both are of order $O(\sqrt{n})$, which leads to $2 \cdot O(\sqrt{n}) \cdot O(\sqrt{n}) = O(n)$.

Hence, in all cases we have $O(n)$ as overall running time of ICB.

3 Concluding remarks

In this paper, we provide a subclass of chordal graphs for which the identifying code problem can be solved in linear time by presenting a linear-time algorithm that finds the identifying code number of a block graph. Our algorithm is a generalization of the linear-time algorithm proposed by Auger [3] for trees and works in a similar way, but it takes into account the identifiable condition for block graphs that is not needed in the case of trees. Hence, the recomposition step of our algorithm is built by defining distinct functions accordingly.

The recursively called algorithm RICB could be easily modified to detect during its execution whether or not B is identifiable. Hence, RICB could test whether K has two vertices v_j, v_ℓ different from v_1 with degree $k-1$ and, if yes, return "not identifiable" and stop.

Furthermore, note that our algorithm ICB could be modified in order to obtain an identifying code of minimum size, just by keeping track of the functions where the minimum values are attained. In addition, if B is a vertex-weighted block graph, the ICB can be easily modified in order to return the minimum weighted identifying code number by just changing in Table 1 the entry with value 1 by the weight corresponding to the vertex.

Finally, it is interesting whether similar ideas could be adapted for graph classes with a similar structure, e.g. for cacti (graphs in which every maximal 2-connected subgraph is an edge or a cycle) or for block-cacti (graphs in which every maximal 2-connected subgraph is a clique or a cycle). In all these cases the procedure should consider different functions in the recomposition step.

References

[1] G. Argiroffo, S. Bianchi, Y. Lucarini, A. Wagler, *Polyhedra associated with identifying codes in graphs*, to appear in: Discrete Applied Mathematics.

[2] G. Argiroffo, S. Bianchi, A. Wagler, *Study of identifying code polyhedra for some families of split graphs*, Lecture Notes in Computer Science 8596 (2014) 13–25.

[3] D. Auger, *Minimal identifying codes in trees and planar graphs with large girth*, European Journal of Combinatorics 31 (2010) 1372–1384.

[4] H.J. Bandelt, H.M. Mulder, *Distance-hereditary graphs*, Journal of Combinatorial Theory, Series B, 41 (1986) 182–208.

[5] N. Bertrand, I. Charon, O. Hudry, A. Lobstein, *Identifying and locating dominating codes on chains and cycles*, Eur. J. Comb. 25 (2004) 969–987.

[6] I. Charon, O. Hudry, A. Lobstein, *Minimizing the size of an identifying or locating-dominating code in a graph is NP-hard.* Theor. Comp. Sc. 290 (2003) 2109–2120.

[7] F. Foucaud, *The complexity of the identifying code problem in restricted graph classes.* Comb. Algorithms (IWOCA 2013), LNCS 8288 (2013) 150–163.

[8] S. Gravier, J. Moncel, *On graphs having a $V\ \{x\}$-set as an identifying code*, Discrete Mathematics 307 (2007) 432–434.

[9] E. Howorka, *On metric properties of certain clique graphs*, Journal of Combinatorial Theory, Series B, 27 (1979) 67–74.

[10] M.G. Karpovsky, K. Chakrabarty, L.B. Levitin, *On a new class of codes for identifying vertices in graphs.* IEEE Trans. Inf. Theory 44 (1998) 599–611.