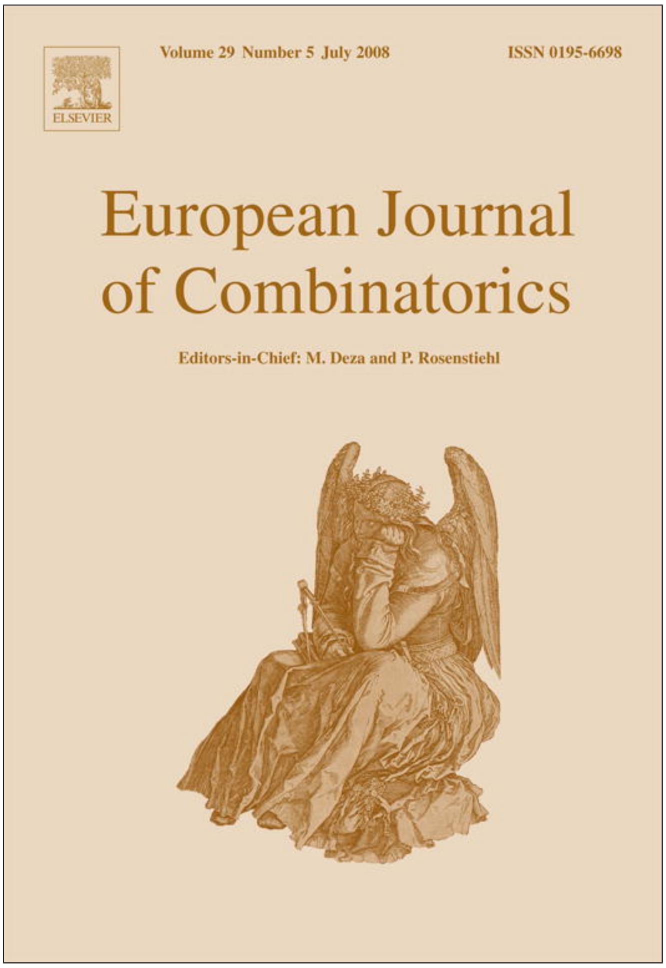
ELSEVIER
Volume 29 Number 5 July 2008
ISSN 0195-6698
European Journal of Combinatorics
Editors-in-Chief: M. Deza and P. Rosenstiehl

Available online at www.sciencedirect.com
ScienceDirect
European Journal of Combinatorics 29 (2008) 1353–1364
European Journal of Combinatorics
www.elsevier.com/locate/ejc

Discriminating codes in (bipartite) planar graphs

Irène Charon[a,b], Gérard Cohen[a,b], Olivier Hudry[a,b], Antoine Lobstein[b,a,1]

[a] *GET, Télécom Paris, 46, rue Barrault, 75634 Paris Cedex 13, France*
[b] *CNRS, LTCI UMR 5141, 46, rue Barrault, 75634 Paris Cedex 13, France*

Received 2 November 2006; accepted 24 May 2007
Available online 27 June 2007

Abstract

Consider a connected undirected bipartite graph $G = (V = I \cup A, E)$, with no edges inside I or A. For any vertex $v \in V$, let $N(v)$ be the set of neighbours of v. A code $C \subseteq A$ is said to be discriminating if all the sets $N(i) \cap C$, $i \in I$, are nonempty and distinct.

We study some properties of discriminating codes in particular classes of bipartite graphs, namely trees and, more generally, (bipartite) planar graphs.

1. Introduction

Let $G = (V = I \cup A, E)$ be a connected undirected bipartite graph, with $E \subseteq \{\{i, a\} : i \in I, a \in A\}$. For any vertex $v \in V$, let $N(v)$ denote the neighbourhood of v and $B(v) = N(v) \cup \{v\}$. Whenever two vertices v_1 and v_2 are neighbours, we say that they *cover* each other. A set $X \subseteq A$ covers a set $Y \subseteq I$ if every vertex in Y is covered by at least one vertex in X.

A *code* C is a nonempty subset of A, whose elements are called *codewords*. For each element $i \in I$, we denote by

$$K_C(i) = N(i) \cap C$$

the set of vertices which are both codewords and neighbours of i. Two vertices i_1 and i_2 with $K_C(i_1) \neq K_C(i_2)$ are said to be *discriminated* by code C.

E-mail addresses: irene.charon@enst.fr (I. Charon), gerard.cohen@enst.fr (G. Cohen), olivier.hudry@enst.fr (O. Hudry), antoine.lobstein@enst.fr (A. Lobstein).
[1] Fax: +33 1 45 81 31 19.

0195-6698/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.ejc.2007.05.006

A code C is called *discriminating* if the sets $K_C(i)$, $i \in I$, are all nonempty and distinct [1]. In other words, all vertices in I must be discriminated and covered by C.

The motivation for these definitions comes from the fact that the sets I and A can be seen as a set of *individuals* and a set of *attributes*, respectively, with an edge between $i \in I$ and $a \in A$ if i owns a; then a discriminating code is a set of attributes which can distinguish between all the individuals (and an individual must own at least one codeword).

This problem can also be expressed in terms of binary matrices: given a matrix, where rows represent individuals and columns attributes, find columns inducing a submatrix with no zero row and no two equal rows.

Discriminating codes are closely related to *(one-)identifying codes* [5], the definition of which can be found in Section 2, and to *(one-)locating–dominating codes* (see [6]), the differences being that here codewords belong to a prescribed subset (namely the set A of attributes), and only a prescribed set of vertices (namely, the set I of individuals) must be identified; see also Remark 3 at the end of Section 4. See [7] for a large bibliography on identifying and locating–dominating codes.

Remark 1. For a given bipartite graph $G = (I \cup A, E)$, there exists a discriminating code $C \subseteq A$ if, and only if,

$$\forall i_1, i_2 \in I (i_1 \neq i_2), \qquad N(i_1) \neq N(i_2).$$

Indeed, if for all $i_1, i_2 \in I$, $N(i_1)$ and $N(i_2)$ are different, then $C = A$ is discriminating. Conversely, if for some $i_1, i_2 \in I$, $N(i_1) = N(i_2)$, then for any code $C \subseteq A$, we have $K_C(i_1) = K_C(i_2)$. Individuals i_1 and i_2 with $N(i_1) = N(i_2)$ are called *twins*, and we can reformulate our remark in the following way: a bipartite graph admits at least one discriminating code if, and only if, it is twin-free.

Obviously, given a bipartite twin-free graph, we shall be interested in finding the smallest possible discriminating code.

In this paper, we investigate the case of trees and planar graphs. In Section 3, we exhibit a polynomial-time (and even linear-time) algorithm finding a smallest discriminating code in a given tree, whereas in general, this problem is NP-hard [1] (for completeness, we give in Section 2 the full proof of this result). In Section 4, we study planar graphs which are optimal in a certain sense.

2. A complexity result

In [1] is sketched the proof that finding the smallest discriminating code is NP-hard in general. More specifically:

Theorem 1. *The following decision problem is NP-complete:*

Name: Discrimination (DISC).
Instance: A bipartite graph $G = (V = I \cup A, E)$, an integer k.
Question: Is there a discriminating code $C \subseteq A$ of size at most k?

We give here a proof which is more detailed than that in [1]. For notions of complexity theory and NP-completeness, we refer the reader to [4].

Since we shall consider two graphs G and G^*, in order to avoid ambiguities we shall use the notation $N_G(v)$ for $v \in G$ and $B_{G^*}(w)$ for $w \in G^*$.

First, we see that DISC $\in$ NP, by observing that, given a set $C \subseteq A$, it is polynomial, with respect to the number of vertices of G, to check whether C is, or is not, a discriminating code (of convenient size).

Next, we polynomially reduce the following NP-complete problem to DISC:

Name: Identification (ID).
Instance: A graph $G^* = (V^*, E^*)$, an integer k^*.
Question: Is there an identifying code of size at most k^* in G^*, i.e., a set $C^* \subseteq V^*$ of size at most k^* such that the sets $B_{G^*}(v)$, $v \in V^*$, are all nonempty and distinct?

It has been proved in [3] that ID is NP-complete. We consider an instance of ID and, starting from G^*, we construct the bipartite graph G in the following way: $I = V^*$, $A = \{B_{G^*}(v) : v \in V^*\}$, $E = \{\{i, B_{G^*}(v)\} : i \in B_{G^*}(v), v \in V^*\}$; we set $k = k^*$.

We claim that there is an identifying code of size at most k in G^* if, and only if, there is a discriminating code of size at most k in G.

If $C^* \subseteq V^*$ is an identifying code in G^*, all the sets $B_{G^*}(v) \cap C^*$, $v \in V^*$, are nonempty and distinct. A fortiori, the sets $B_{G^*}(v)$ are all distinct; consequently, the set $C = \{B_{G^*}(v) : v \in C^*\} \subseteq A$ has $|C^*|$ elements. Moreover, C is a discriminating code in G: for $i \in I$,

$$N_G(i) \cap C = \{B_{G^*}(v) : i \in B_{G^*}(v), v \in C^*\},$$

or

$$N_G(i) \cap C = \{B_{G^*}(v) : v \in B_{G^*}(i) \cap C^*\}.$$

Since the sets $B_{G^*}(i) \cap C^*$, $i \in V^*$, are nonempty and distinct, the same is true for the sets $N_G(i) \cap C$, $i \in I$.

Conversely, assume that we have a discriminating code $C \subseteq A$ in G; there exists $C^* \subseteq V^*$ such that $C = \{B_{G^*}(v) : v \in C^*\}$, and we claim that C^* is an identifying code in G^*. If not, then either a set $B_{G^*}(v_0) \cap C^*$ is empty or two sets $B_{G^*}(v_1) \cap C^*$, $B_{G^*}(v_2) \cap C^*$ are equal. In the former case, v_0, seen as a vertex in I, is such that $N_G(v_0) \cap C$ is empty; in the latter, $N_G(v_1) \cap C = N_G(v_2) \cap C$. In both cases, we get a contradiction. Finally, C^* and C obviously have the same size.

We have therefore transformed ID to DISC in such a way that there is a positive instance for ID if, and only if, there is a positive instance for DISC. This, together with the obvious polynomiality of the transformation and the membership to NP, proves that DISC is NP-complete. $\triangle$

However, if we restrict ourselves to trees, we can answer the question in DISC (in a constructive way) in *polynomial* (and even *linear*) time, as we show in the next section.

From now on, the graphs considered are always bipartite, connected and twin-free.

3. A polynomial algorithm for trees

Let $T = (V = I \cup A, E)$ be a tree with n vertices. A leaf $\ell \in V$ is a vertex with only one neighbour, which we denote by $n(\ell)$. If $\ell \in I$, then $n(\ell) \in A$ and there is no other leaf whose neighbour is $n(\ell)$, since T is twin-free.

We can also observe that if $\ell \in A$, then $n(\ell) \in I$ and every leaf with neighbour $n(\ell)$ is an attribute; if there are several leaves with neighbour $n(\ell)$, it is then obvious that among them, at most one will belong to a discriminating code with minimum size.

First we show how we can either solve the problem or transform it into one or several smaller problems; then we show that this approach indeed leads to an algorithm which is polynomial with respect to n. Finally we mention that a linear algorithm exists.

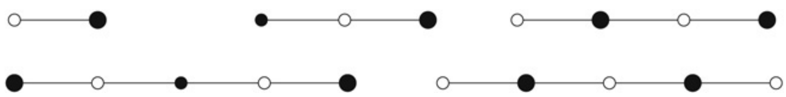

Fig. 1. Different paths of length up to four, with optimal discriminating codes. White circles are individuals, black circles are attributes, with a larger diameter when they are codewords.

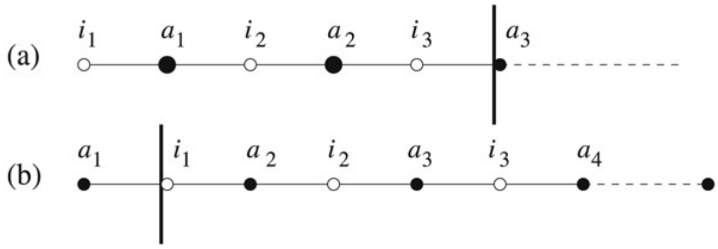

Fig. 2. How to cut the paths of length at least 5.

3.1. Paths

For T a path of length (i.e., number of *edges*) at most four, we give in Fig. 1 all the possibilities. We now study paths of length $L \geq 5$, and distinguish between two cases.

(a) There is an individual at one end of the path, so we can write $T = i_1, a_1, i_2, a_2, i_3, a_3, \ldots$; see Fig. 2(a).

Because i_1 must be covered by a codeword and i_1 and i_2 must be discriminated, necessarily both a_1 and a_2 belong to any discriminating code. Now i_3 is covered by a_2 and will be discriminated from i_4, whatever the status of a_3 and a_4 will be. So, remembering that a_1 and a_2 are codewords, we can cut the path between i_3 and a_3 and see that the problem of searching for an optimal code in the whole path is equivalent to the same problem in the reduced path.

(b) Both ends of the path are attributes, so $L \geq 6$ is odd and we can write $T = a_1, i_1, a_2, i_2, a_3, i_3, a_4, \ldots, a_{1+L/2}$; see Fig. 2(b).

Suppose that a_1 belongs to a minimal code in T; then it is easy to see that this optimal code contains exactly one of a_2 and a_3; if it is a_2 (respectively, a_3), then another optimal code can be obtained by replacing a_1 by a_3 (respectively, a_2). This shows that, remembering that a_1 will not be a codeword, we can cut the path between a_1 and i_1, and, as in Case (a), search for an optimal code in the remaining path.

This shows how to progressively reduce the problem to the paths given in Fig. 1.

3.2. Threads and branchings

From now on, we assume that T is not a path, which means that there is at least one vertex with degree at least three.

We call *thread* any path such that (a) one of its two extremities is a leaf, which we call the *end* of the thread, and the other is a vertex of degree at least three, which we call the *origin* of the thread, and (b) all in-between vertices have degree two. We distinguish between three cases.

(a) There exists in T a thread of length at least five whose end is an individual i_1 and whose origin w is either an individual or an attribute; see Fig. 3(a).

As in the case of paths, the attributes a_1 and a_2 belong to any discriminating code in T since i_1 must be covered by a codeword and i_1 and i_2 must be discriminated. It is then easy to observe, as previously for paths, that the problem of finding an optimal code is the same for in T (where we take a_1 and a_2 as codewords) and for in T deprived of the vertices i_1, a_1, i_2, a_2 and i_3.

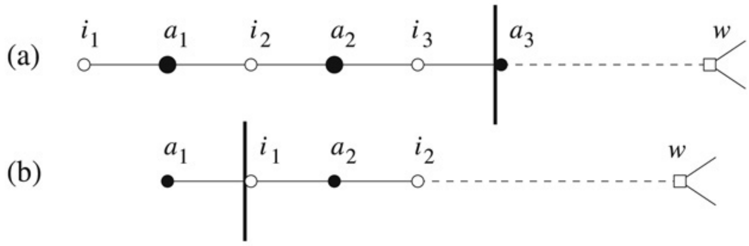

Fig. 3. How to cut threads. Squares can be individuals or attributes.

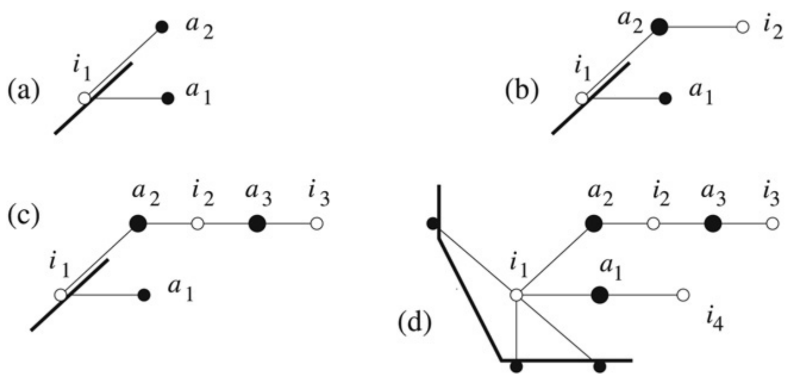

Fig. 4. Each branching is an individual.

(b) There exists in T a thread of length at least three whose end is an attribute a_1 and whose origin w is either an individual or an attribute; see Fig. 3(b).

We claim that there exists an optimal discriminating code not containing a_1. Assume that C is an optimal discriminating code. If $a_1 \notin C$, we are done; if $a_1 \in C$, we remove a_1 from C: the resulting code is not discriminating. This is so either because now i_1 is not covered by any codeword, or because i_1 and i_2 are not discriminated. In the former case, $a_2 \notin C$ and i_2 is covered by another codeword; we can therefore replace a_1 by a_2 in C and still have a discriminating code. In the latter case, the only codeword covering i_2 is a_2; in C, we can therefore replace a_1 by any attribute which is neighbour to i_2 and different from a_2 and still have a discriminating code. In both cases, our claim is true. As a consequence, we can, keeping in mind that a_1 will not be a codeword, cut the thread between a_1 and i_1, and search for an optimal code in the remaining tree.

(c) All threads whose end is an attribute have length at most two and all threads whose end is an individual have length at most four.

We call *branching* any vertex of degree at least three which is the origin of at least two threads. Branchings exist in T because T is not a path. We distinguish between two cases for branchings: individuals or attributes.

(i) The branching is an individual. Its threads can have length 1, 2 or 4: length 3 is impossible since the end of the thread would be an attribute. We have two subcases.

(i1) The branching has a thread of length one.

If there are two threads of length one, since their ends are attributes, there is one which needs not be a codeword and can be removed from the current tree; see Fig. 4(a).

If there is a thread of length two, see Fig. 4(b), then necessarily a_2 is a codeword, and if a_1 belongs to an optimal code, we can replace it by an attribute, other than a_2 and neighbour of i_1 (remember that i_1 has degree at least three); as before, we see that we can, keeping in mind that a_1 will not be a codeword, delete a_1 and search for an optimal code in the remaining tree.

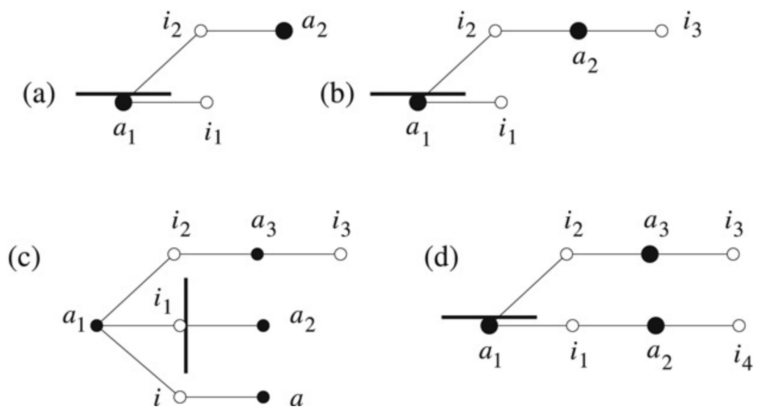

Fig. 5. Each branching is an attribute.

Finally, if there is a thread of length four, see Fig. 4(c), necessarily a_2 and a_3 belong to any discriminating code, and we can as before delete a_1.

(i2) All the threads of the branching have length two or four; cf. Fig. 4(d).

Obviously, a_1, a_2 and a_3, and, more generally, all the attributes on all the threads, belong to any discriminating code. Therefore, if we remove all the threads (including the branching), then in the remaining forest, any optimal code will give, together with the attributes on the threads, an optimal code in T. This completes Case (i).

(ii) The branching is an attribute. Its threads can have length 1, 2 or 3: length 4 is impossible since the end of the thread would be an attribute. We have already noticed that there can be at most one thread of length one. We have three subcases.

(ii1) There is a thread of length one.

If there is a thread of length two, see Fig. 5(a), then a_1 and a_2 belong to any discriminating code and, using the same arguments as before, we see that we can remove i_2 and a_2.

If there is a thread of length three, see Fig. 5(b), then again a_1 and a_2 must be codewords, and we can remove i_2, a_2 and i_3.

(ii2) The smallest thread has length two; cf. Fig. 5(c).

We claim that there exists an optimal discriminating code not containing a_2. Assume that C is an optimal discriminating code. If $a_2 \notin C$, we are done; if $a_2 \in C$, we remove a_2 from C: the resulting code is not discriminating. This is so either because now i_1 is not covered by any codeword, or because i_1 is not discriminated from some (unique) individual i. In the former case, $a_1 \notin C$ and all its neighbours other than i_1 are covered by some other codeword(s); we can therefore replace a_2 by a_1 in C and still have a discriminating code. In the latter case, $a_1 \in C$ is the only codeword covering i_1 and i; since there is no thread of length one, i has a neighbour $a \notin C$ and we can replace a_2 by a in C, proving our claim. Consequently, we can remove a_2 from the tree.

(ii3) All threads have length three; see Fig. 5(d).

Then a_1, a_2 and a_3, and more generally all the attributes on all the threads, belong to any discriminating code, and we can remove the vertices i_2, a_3 and i_3 from T.

This completes Case (ii), and Case (c) is treated.

3.3. Summary and conclusion

From Sections 3.1 and 3.2, it is now easy to see how an algorithm outputting an optimal discriminating code can work.

If T is a path of length at most four, we use Fig. 1. If T is a path of length at least five, we reduce it following Cases (a) and (b) in Section 3.1, cf. Fig. 2, until its length is at most four. By the way, we observe that a discriminating code needs to contain approximately two out of three attributes.

Now we assume that T is not a path. There are special induced subgraphs which we called threads. We reduce the lengths of these threads, cf. Cases (a) and (b) in Section 3.2 and Fig. 3, until we are in Case (c): any thread whose end is an attribute has length at most two and any thread whose end is an individual has length at most four. Then we investigate the vertices which we called branchings and still reduce the current tree; see Figs. 4 and 5. At this stage, the tree may be disconnected; we then operate on each connected component. At each step, noting whether attributes that were removed are codewords or not, the search for an optimal code in the reduced tree (or forest) will give an optimal code in the whole tree T.

At each step, we perform polynomial-time operations (with respect to n, the number of vertices of T) and decrease n, so that the algorithm sketched above will stop after a polynomial time. With careful investigation on which data structures to use, we could even show that this algorithm is linear with respect to n. We refer the reader to [2] for a similar work.

4. Planar graphs

In this section, every graph is connected, bipartite, twin-free and planar. Note that a planar graph may admit several plane embeddings, in which the lengths of the faces may vary.

Rather than searching for the smallest discriminating codes in planar graphs, we address the following issue: we fix $|A|$, the number of attributes, and construct a connected, bipartite, twin-free, planar graph $G = (V = A \cup I, E)$ with the maximum possible number, $|I|$, of individuals which can be discriminated by the attributes.

From this angle, we shall denote by $\alpha(G)$ the number of attributes and by $\beta(G)$ the number of individuals: $\alpha(G) = |A|$, $\beta(G) = |I|$. The graph G is said to be *optimal* if no graph G' is such that $\alpha(G) = \alpha(G')$ and $\beta(G) < \beta(G')$.

We shall denote by $\mathcal{G}_2$ the set of graphs such that any *individual* has degree at least two, and say that a graph $G \in \mathcal{G}_2$ is *optimal in* $\mathcal{G}_2$ if no graph $G' \in \mathcal{G}_2$ is such that $\alpha(G) = \alpha(G')$ and $\beta(G) < \beta(G')$.

Theorem 2. *If $G \in \mathcal{G}_2$ has $\alpha(G) \geq 4$ attributes, then G is optimal if, and only if, in every plane embedding of G, each face has length four and contains one individual of degree two and one individual of degree three.*

Proof. The proof is constructive and shows how to transform a graph in $\mathcal{G}_2$ into an optimal graph in $\mathcal{G}_2$.

During the process of our transformation, it may happen that we delete edges, but the resulting graph will never be disconnected.

Lemmas 3–9 will be proved in the course of the proof of Theorem 2.

Lemma 3. *If $G \in \mathcal{G}_2$ has at least three attributes and admits a plane embedding with a face which is not delimited by an elementary cycle, then to G we can add one individual and obtain a new graph belonging to $\mathcal{G}_2$.*

Proof. We assume that F is a face in G which is not delimited by an elementary cycle. If F is not delimited by a cycle, then G has only one face and is a tree. It is rather straightforward to

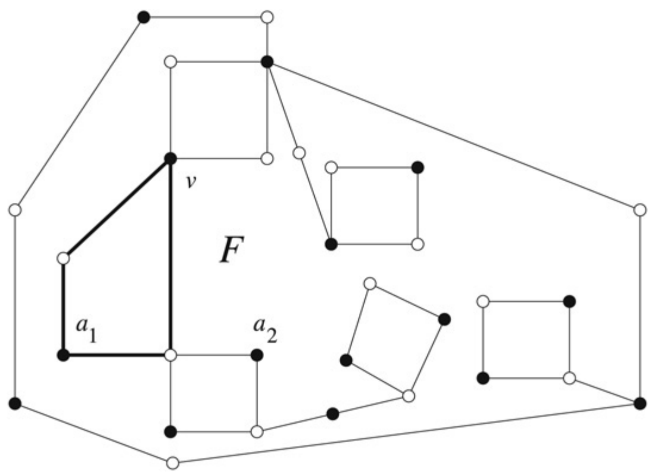

Fig. 6. An example of a face delimited by a cycle $\mathcal{C}$ containing an elementary cycle $\mathcal{C}'$, represented with bold edges. The vertex v may be an individual or an attribute.

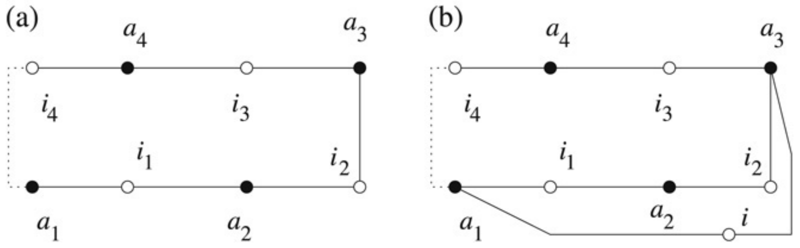

Fig. 7. A face of length at least eight.

observe that any tree in $\mathcal{G}_2$ with at least four vertices is such that we can add one individual and obtain a new graph belonging to $\mathcal{G}_2$. So from now on we assume that F is delimited by a cycle $\mathcal{C}$, from which we can extract an elementary cycle $\mathcal{C}'$. Let $\mathcal{C}'' = \mathcal{C} \setminus \mathcal{C}'$ and let v be a vertex connecting $\mathcal{C}'$ and $\mathcal{C}''$ (v is not necessarily unique); see Fig. 6. Since G is bipartite, $\mathcal{C}'$ contains at least four vertices, and thus at least one attribute, a_1, which is not equal to v. We claim on the other hand that $\mathcal{C}''$ contains at least one attribute, a_2; first $\mathcal{C}''$ is not empty; second if $\mathcal{C}''$ contains only individuals, then these individuals, which have at least a second neighbour since $G \in \mathcal{G}_2$, are linked to attributes not belonging to $\mathcal{C}''$, and we can see that, as a consequence, F would not be delimited by $\mathcal{C}$. It is then easy to see that we can add one individual with neighbourhood $\{a_1, a_2\}$, which yields a new graph in $\mathcal{G}_2$, and ends the proof of Lemma 3. △

Definition 1. We denote by (P1) the following property of a graph $G \in \mathcal{G}_2$: in every plane embedding of G, any face is delimited by an elementary cycle.

Lemma 4. *If $G \in \mathcal{G}_2$ satisfies* (P1) *and admits a plane embedding with a face of length greater than or equal to* 8*, then to G we can add one individual and obtain a new graph belonging to $\mathcal{G}_2$.*

Proof. Let $\phi = a_1, i_1, a_2, i_2, a_3, i_3, a_4, i_4, \ldots$ be a face of length 8 or more; see Fig. 7(a). If there is no individual with neighbourhood $\{a_1, a_3\}$, we add such an individual inside ϕ. If such an individual, i, already exists outside ϕ, see Fig. 7(b), then no path can exist between a_2 and a_4 because such a path would cross either the path a_1, i, a_3 or the face ϕ. We can therefore add one individual with neighbourhood $\{a_2, a_4\}$ inside ϕ, which yields a new graph in $\mathcal{G}_2$, and ends the proof of Lemma 4. △

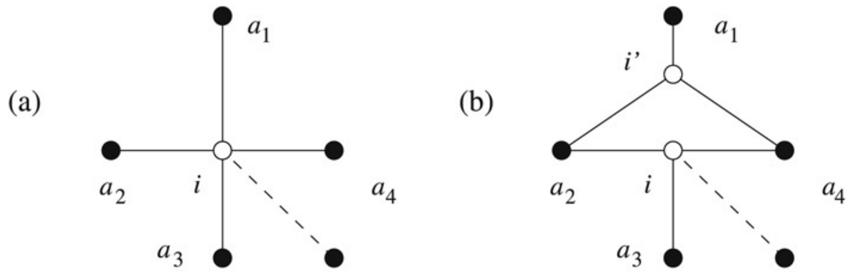

Fig. 8. An individual i of degree at least four.

Definition 2. We denote by (P2) the following property of a graph $G \in \mathcal{G}_2$: in every plane embedding of G, any face has length at most six — remember that all cycles, and hence faces, are even.

Lemma 5. *If $G \in \mathcal{G}_2$ satisfies* (P1) *and* (P2) *and has an individual of degree at least four, then we can delete one edge, add one individual and obtain a new graph belonging to $\mathcal{G}_2$.*

Proof. Let a_1, a_2, a_3, a_4 be four neighbours of an individual i of degree at least four in G; see Fig. 8(a). At most one of the following two possibilities can occur: there is an individual, other than i, linked to a_1 and a_3; there is an individual, other than i, linked to a_2 and a_4. Therefore, without loss of generality, we can assume that i is the only individual linked to a_2 and a_4.

Now we delete the edge $\{a_1, i\}$ and add an individual i' with neighbourhood $\{a_1, a_2, a_4\}$, which, thanks to our assumption on a_2 and a_4, yields a graph which is still twin-free and belongs to $\mathcal{G}_2$; see Fig. 8(b). This ends the proof of Lemma 5. △

Definition 3. We denote by (P3) the following property of a graph $G \in \mathcal{G}_2$: any individual has degree two or three.

Lemma 6. *Consider a graph $G \in \mathcal{G}_2$ which satisfies* (P1)–(P3) *and has at least four attributes. If G admits a plane embedding with a face of length six, then we can replace one edge, add one individual and obtain a new graph belonging to $\mathcal{G}_2$.*

Proof. Let $\phi = a_1, i_1, a_2, i_2, a_3, i_3, a_1$ be a face of length six. If there is no individual with neighbourhood $\{a_1, a_2, a_3\}$, we can add such an individual inside ϕ. So we assume that such an individual, i, already exists outside ϕ; see Fig. 9(a). We distinguish between two cases.

(i) There is an individual with degree three belonging to ϕ, say i_1. Let a be the third neighbour of i_1; see Fig. 9(b). Then no individual can be the neighbour of both a_3 and a. We replace the edge $\{i_1, a_1\}$ by the edge $\{i_1, a_3\}$, and we add the individual i' with neighbourhood $\{a, a_3\}$; see Fig. 9(c). The graph thus constructed is still twin-free and in $\mathcal{G}_2$.

(ii) All individuals in ϕ have degree two. Because (P3) is satisfied, $N(i) = \{a_1, a_2, a_3\}$; cf. Fig. 9(a). If the only face containing the path a_1, i, a_3 (respectively, a_3, i, a_2 and a_1, i, a_2) has length four, then this face is a_1, i, a_3, i_3, a_1 (respectively, a_3, i, a_2, i_2, a_3 and a_1, i, a_2, i_1, a_1). If these three faces have simultaneously length four, then we can have no vertex other than the seven vertices given in Fig. 9(a), which contradicts the assumption that G has at least four attributes. Therefore, at least one of them is of length six and contains the individual i, which has degree three, and we are back to Case (i), which ends the proof of Lemma 6. △

Lemma 7. *Consider a graph $G \in \mathcal{G}_2$ which satisfies* (P1)–(P3). *If G admits a plane embedding with a face of length four with two individuals of degree three, then we can replace two edges, add one individual and obtain a new graph belonging to $\mathcal{G}_2$.*

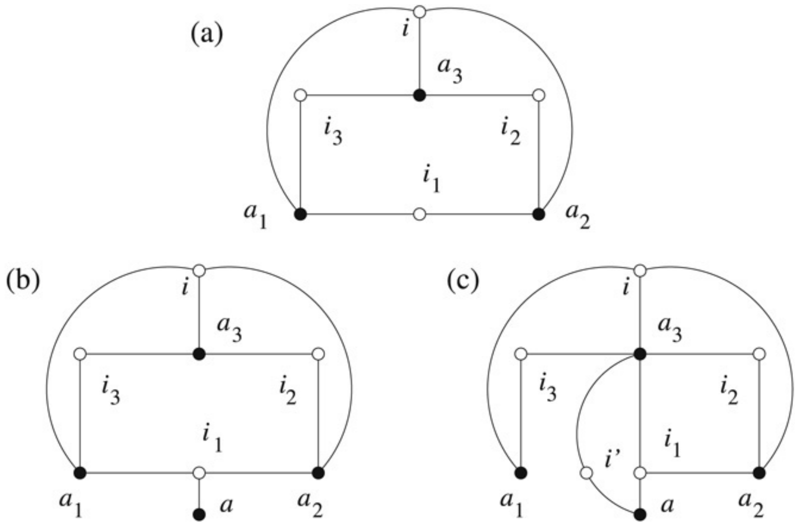

Fig. 9. A face of length six.

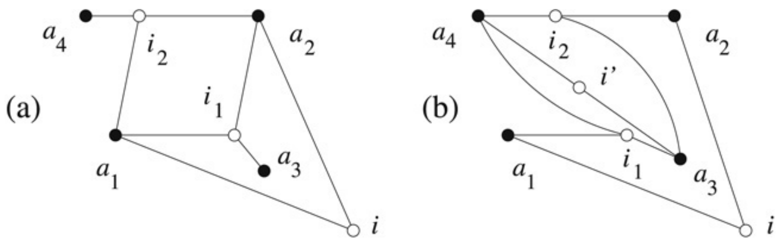

Fig. 10. A face of length four with two individuals of degree three.

Proof. Let $\phi = a_1, i_1, a_2, i_2, a_1$ be a face of length four, with $N(i_1) = \{a_1, a_2, a_3\}$ and $N(i_2) = \{a_1, a_2, a_4\}$. As before, we can assume that there is already, outside ϕ, an individual i with neighbourhood $\{a_1, a_2\}$; see Fig. 10(a). This in turn makes impossible the existence of an individual linked to both a_3 and a_4. Now we delete the edges $\{i_1, a_2\}$, $\{i_2, a_1\}$, add the edges $\{a_3, i_2\}$ and $\{i_1, a_4\}$, and create a new individual i' with $N(i') = \{a_3, a_4\}$; see Fig. 10(b). The graph thus constructed is still twin-free and in $\mathcal{G}_2$, which ends the proof of Lemma 7. △

Lemma 8. *In an optimal graph $G \in \mathcal{G}_2$ with at least four attributes, in every plane embedding, every face has length four and contains one individual of degree two and one individual of degree three.*

Proof. Use Lemmas 3–7 and the fact that in a face of length four, the two individuals cannot both have degree two (they would be twins). △

It remains to show that any graph $G \in \mathcal{G}_2$ with at least four attributes, for which, in every plane embedding, all faces have length four and contain one individual of degree two and one of degree three, is optimal. To do so, we prove that all such graphs, with a given number of attributes, $\alpha \geq 4$, have the same number of individuals, denoted by $\beta(\alpha)$.

Lemma 9. *Let $G \in \mathcal{G}_2$ be a graph with $\alpha \geq 4$ attributes, for which, in every plane embedding, all faces have length four and contain one individual of degree two and one individual of degree three. Then $\beta(G) = \beta(\alpha) = 5\alpha - 10$.*

Proof. Let n be the number of vertices in G, m the number of edges and Φ the number of faces. We have

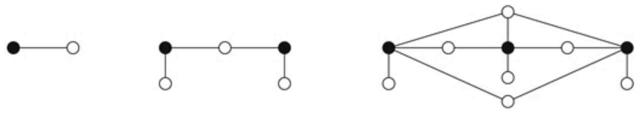

Fig. 11. Optimal constructions for one, two and three attributes: $\beta(1) = 1$, $\beta(2) = 3$, $\beta(3) = 7$.

$$n = \alpha + \beta(\alpha), \tag{1}$$

and it is well known that

$$n + \Phi = m + 2. \tag{2}$$

Let β_2 and β_3 be the numbers of individuals with degrees two and three, respectively; we know that

$$\beta(\alpha) = \beta_2 + \beta_3, \tag{3}$$

and trivially

$$m = 2\beta_2 + 3\beta_3. \tag{4}$$

Moreover, since there is one individual with degree two (respectively, three) on each face, if we count all these individuals on all faces, then we count them twice (respectively, thrice); so

$$\Phi = 2\beta_2 = 3\beta_3. \tag{5}$$

Combining (3) and (5), we obtain

$$\beta_2 = \frac{3\beta(\alpha)}{5}; \qquad \beta_3 = \frac{2\beta(\alpha)}{5}, \tag{6}$$

which, by (4) and (5) yields

$$m = \frac{12\beta(\alpha)}{5}; \qquad \Phi = \frac{6\beta(\alpha)}{5}. \tag{7}$$

Now in (2), we replace n, Φ and m by their values given by (1) and (7), and we obtain

$$\alpha + \beta(\alpha) + \frac{6\beta(\alpha)}{5} = \frac{12\beta(\alpha)}{5} + 2,$$

from which the result in Lemma 9 follows. △

As mentioned previously, this is sufficient for completing the proof of Theorem 2. △

Corollary 10. *Let G be a graph with $\alpha \geq 4$ attributes. Then $\beta(G) \leq 6\alpha - 10$, and the optimal graphs have $\beta(\alpha) = 6\alpha - 10$ individuals.*

Proof. Use the graphs in $\mathcal{G}_2$ and to each attribute link one individual which is not linked to any other attribute. △

The cases $\alpha \leq 3$ are easily solved by the constructions given in Fig. 11, which are optimal since they meet the trivial bound $\beta(\alpha) \leq 2^\alpha - 1$.

Remark 2. There is a bijection between the set of optimal graphs and the set of triangulated (planar) graphs. Indeed, starting from an optimal graph, we can carry out the following three operations:

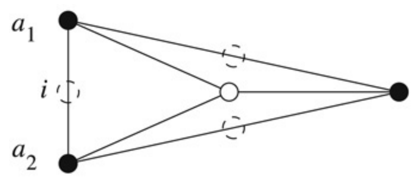

Fig. 12. A tile with an individual of degree three. Dashed individuals were removed.

(a) We delete the individuals of degree one.
(b) We replace each individual i of degree two and its edges $\{i, a_1\}$, $\{i, a_2\}$ by one edge $\{a_1, a_2\}$; we then obtain a tiling by tiles of the form given in Fig. 12.
(c) We delete the individuals of degree three (and their edges) and obtain a triangulated graph.

It is easy to check that this application from the set of optimal graphs with α attributes into the set of triangulated graphs of order α is bijective.

Remark 3. The result of Corollary 10 can also be obtained from [6], where locating–dominating codes are constructed for bipartite planar graphs, using the inverse of the application described in Remark 2: if by construction all the attributes are the elements of a discriminating code on the one hand, and if on the other hand there are no edges inside the set of elements of a locating–dominating code or inside its complement in V, then the two notions coincide.

References

[1] E. Charbit, I. Charon, G. Cohen, O. Hudry, Discriminating codes in bipartite graphs, Electronic Notes in Discrete Mathematics 26 (2006) 29–35.
[2] I. Charon, S. Gravier, O. Hudry, A. Lobstein, M. Mollard, J. Moncel, A linear algorithm for minimum 1-identifying codes in oriented trees, Discrete Applied Mathematics 154 (2006) 1246–1253.
[3] G. Cohen, I. Honkala, A. Lobstein, G. Zémor, On identifying codes, in: DIMACS Series in Discrete Mathematics and Theoretical Computer Science, in: Proceedings of the DIMACS Workshop on Codes and Association Schemes, vol. 56, 2001, pp. 97–109.
[4] M.R. Garey, D.S. Johnson, Computers and Intractability, A Guide to the Theory of NP-Completeness, Freeman, New York, 1979.
[5] M.G. Karpovsky, K. Chakrabarty, L.B. Levitin, On a new class of codes for identifying vertices in graphs, IEEE Transactions on Information Theory 44 (1998) 599–611.
[6] D.F. Rall, P.J. Slater, On location–domination numbers for certain classes of graphs, Congressus Numerantium 45 (1984) 97–106.
[7] http://www.infres.enst.fr/~lobstein/bibLOCDOMetID.html.