Theoretical Computer Science 626 (2016) 1–12

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

More results on the complexity of identifying problems in graphs

Olivier Hudry [a,*], Antoine Lobstein [b]

[a] Institut Télécom, Télécom ParisTech & CNRS, LTCI UMR 5141, 46, rue Barrault, 75634 Paris Cedex 13, France
[b] CNRS, LTCI UMR 5141 & Institut Télécom, Télécom ParisTech, 46, rue Barrault, 75634 Paris Cedex 13, France

ARTICLE INFO

Article history:
Received 3 June 2015
Received in revised form 16 November 2015
Accepted 16 January 2016
Available online 27 January 2016
Communicated by C. Kaklamanis

Keywords:
Graph theory
Complexity
Complexity classes
Polynomial hierarchy
NP-completeness
Hardness
Identifying codes
Twin-free graphs

ABSTRACT

We investigate the complexity of several problems linked with identification in graphs; for instance, given an integer $r \geq 1$ and a graph $G = (V, E)$, the existence of, or search for, optimal r-identifying codes in G, or optimal r-identifying codes in G containing a subset of vertices $X \subset V$. We locate these problems in the complexity classes of the polynomial hierarchy.

1. Introduction and preliminary results

Following [17], which investigates the complexity of Slater's problems in tournaments, our goal in this paper is to study the algorithmic complexity of different variants of the identifying problem in graphs.

In [18], we do the same work for domination problems.

1.1. Outline of the paper

In Subsection 1.2, we present the necessary notation and definitions about identifying codes; Subsection 1.3 gives preliminary results on identifying codes. In Section 2, we present seven problems, decision, optimization or search problems, related to identification, we give some known results, before we motivate our research and give our own development. We shall provide the necessary notions of complexity as we go along. The conclusion recapitulates our results.

1.2. Definitions and notation

We first give the necessary definitions and notation for identification in graphs; see the seminal paper [20], and also [21] for a large bibliography.

* Corresponding author.
E-mail addresses: hudry@telecom-paristech.fr (O. Hudry), lobstein@telecom-paristech.fr (A. Lobstein).

http://dx.doi.org/10.1016/j.tcs.2016.01.021
0304-3975/

We shall denote by $G = (V, E)$ a finite, simple, undirected graph with vertex set V and edge set E, where an *edge* between $x \in V$ and $y \in V$ is indifferently denoted by xy or yx. The *order* of the graph is its number of vertices, $|V|$. A *path* $P_k = x_1x_2 \dots x_k$ is a sequence of k distinct vertices x_i, $1 \le i \le k$, such that x_ix_{i+1} is an edge for $i \in \{1, 2, \dots, k-1\}$. The *length* of P_k is its number of edges, $k-1$.

A graph G is called *connected* if for any two vertices x and y, there is a path between them; it is called *disconnected* otherwise. In a connected graph G, we can define the *distance* between any two vertices x and y, denoted by $d_G(x, y)$, as the length of any shortest path between x and y, since at least one such path exists. This definition can be extended to disconnected graphs, using the convention that $d_G(x, y) = +\infty$ if no path exists between x and y. The subscript G can be dropped when there is no ambiguity.

For an integer $k \ge 2$, the *k-th transitive closure*, or *k-th power* of $G = (V, E)$ is the graph $G^k = (V, E^k)$ defined by $E^k = \{uv : u \in V, v \in V, d_G(u, v) \le k\}$.

For any vertex $v \in V$, the *open neighbourhood* $N(v)$ *of* v consists of the set of vertices adjacent to v, i.e., $N(v) = \{u \in V : uv \in E\}$; the *closed neighbourhood of* v is $B_1(v) = N(v) \cup \{v\}$. This notation can be generalized to any integer $r \ge 0$ by setting

$$B_r(v) = \{x \in V : d(x, v) \le r\}.$$

For $X \subseteq V$, we denote by $B_r(X)$ the set of vertices within distance r from X:

$$B_r(X) = \cup_{x \in X} B_r(x).$$

Two vertices x and y such that $B_r(x) = B_r(y)$, $x \ne y$, are called *r-twins*. If G has no r-twins, we say that G is *r-twin-free*. Whenever two vertices x and y are such that $x \in B_r(y)$ (which is equivalent to $y \in B_r(x)$), we say that x and y *r-cover* or *r-dominate* each other; note that every vertex r-covers itself. A set W is said to r-cover a set Z if every vertex in Z is r-covered by at least one vertex of W. When three vertices x, y, z are such that $z \in B_r(x)$ and $z \notin B_r(y)$, we say that z *r-separates* x and y in G (note that $z = x$ is possible). A set of vertices is said to r-separate x and y if it contains at least one vertex which does.

A *code* C is simply a subset of V, and its elements are called *codewords*. For each vertex $v \in V$, we denote the set of codewords r-covering v by $I_{G,C,r}(v)$, or, when there is no ambiguity on G, by $I_{C,r}(v)$:

$$I_{G,C,r}(v) = I_{C,r}(v) = B_r(v) \cap C.$$

We say that C is an *r-dominating* code in G if all the sets $I_{C,r}(v)$, $v \in V$, are nonempty; in other words, every vertex is r-dominated by C. We say that C is an *r-identifying* code if all the sets $I_{C,r}(v)$, $v \in V$, are nonempty and distinct: in other words, every vertex is r-covered by C, and every pair of vertices is r-separated by C. It is quite easy to observe that a graph G admits an r-identifying code if and only if G is r-twin-free; this is why r-twin-free graphs are also called *r-identifiable*. When G is r-twin-free, we denote by $i_r(G)$ the smallest cardinality of an r-identifying code in G, and call it the *r-identification number* of G. Any r-identifying code C such that $|C| = i_r(G)$ is said to be *optimal*.

1.3. Some useful facts on identification

In the sequel, we shall need the following results on identification.

Lemma 1. *Let $r \ge 1$ be an integer and G be a graph. If C is an r-identifying code in G, then any code $S \supseteq C$ also is.*

Proof. When we add the elements of $S \setminus C$ to the adequate sets $I_{C,r}(v)$, these new sets $I_{S,r}(v)$ are still nonempty and distinct, and distinct from the sets with no addition (those such that $I_{S,r}(v) = I_{C,r}(v)$). □

Lemma 2. *Let $r \ge 2$ be an integer and $G = (V, E)$ be a graph. A code C is 1-identifying in G^r, the r-th power of G, if and only if it is r-identifying in G.*

Proof. For every vertex $v \in V$, we have:

$$I_{G,C,r}(v) = \{c \in C : d_G(v, c) \le r\} = \{c \in C : d_{G^r}(v, c) \le 1\} = I_{G^r,C,1}(v). \qquad \square$$

Lemma 3. *Let $G = (V, E)$ be a 1-twin-free graph. For a given set of vertices $A = \{\alpha_1, \dots, \alpha_k\} \subseteq V$, we construct the following graph $G_A = (V_A, E_A)$, which depends on A (see Fig. 1):*

$$V_A = V \cup V_A^*, \text{ with } V_A^* = \cup_{1 \le j \le k} V_j^* \text{ and } V_j^* = \{\beta_{j,1}, \beta_{j,2}, \delta_j, \lambda_j\},$$

$$E_A = E \cup \{\alpha_j\beta_{j,1}, \beta_{j,1}\beta_{j,2}, \beta_{j,1}\delta_j, \beta_{j,1}\lambda_j, \beta_{j,2}\delta_j, \beta_{j,2}\lambda_j : 1 \le j \le k\},$$

where for $j \in \{1, \dots, k\}$, none of the vertices $\beta_{j,1}, \beta_{j,2}, \delta_j, \lambda_j$ belongs to V.

Then $A \subseteq V$ is included in at least one optimal 1-identifying code in G if and only if $i_1(G) = i_1(G_A) - 2|A|$.

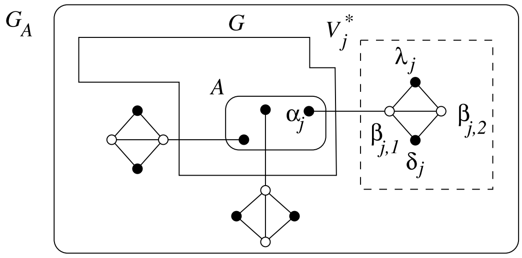

Fig. 1. The graph G_A; black vertices must belong to any 1-identifying code in G_A.

Proof. In the sequel, we denote the set $\{1, \ldots, k\}$ by J_k.

The graph G_A, which is represented in Fig. 1, is constructed in such a way that if C_A is a 1-identifying code in G_A, then, for $j \in J_k$, $\{\alpha_j, \delta_j, \lambda_j\} \subseteq C_A$, because α_j (respectively, δ_j, λ_j) is the only vertex 1-separating $\beta_{j,1}$ and $\beta_{j,2}$ (respectively, $\beta_{j,2}$ and λ_j, $\beta_{j,2}$ and δ_j).

In particular, $A \subset (C_A \cap V)$, and $|C_A \cap V_A^*| \geq 2k$, which implies that $|C_A \cap V| \leq |C_A| - 2k$.

(a) Assume that A is included in at least one optimal 1-identifying code C in G. Then $C_A = C \cup \{\delta_j, \lambda_j : j \in J_k\}$ is a 1-identifying code in G_A, and

$$i_1(G_A) \leq |C_A| = |C| + 2k = i_1(G) + 2k. \tag{1}$$

On the other hand, among all the optimal 1-identifying codes in G_A, consider one, say C_A, which minimizes $|C_A \cap V^*|$. We have already observed that $A \subset (C_A \cap V)$ and $|C_A \cap V| \leq |C_A| - 2k = i_1(G_A) - 2k$. We are going to prove that $C_A \cap V$ is a 1-identifying code in G, which will imply that

$$i_1(G) \leq |C_A \cap V| \leq i_1(G_A) - 2k,$$

and will give, together with (1), the desired equality $i_1(G_A) = i_1(G) + 2k$.

Assume first that $\beta_{j,1} \in C_A$ for some $j \in J_k$.

This vertex is useless as far as covering is concerned, since we know that necessarily α_j, δ_j and λ_j are codewords, and these three vertices 1-cover all the vertices 1-covered by $\beta_{j,1}$.

Next, what is the separating effect of $\beta_{j,1}$ in G_A? It can only 1-separate α_j from some vertices in $V \setminus \{\alpha_j\}$, say $y_1, \ldots, y_m$. If α_j is 1-separated from all the y_i's by $C_A \setminus \{\beta_{j,1}\}$, then $\beta_{j,1}$ is useless, and C_A is not optimal. So we assume that there are ℓ vertices, say $y_1, \ldots, y_\ell$, in $V \setminus \{\alpha_j\}$, $m \geq \ell > 0$, which are 1-separated from α_j only by $\beta_{j,1}$:

$$I_{G_A,C_A,1}(y_1) = I_{G_A,C_A,1}(y_2) = \ldots = I_{G_A,C_A,1}(y_\ell) = I_{G_A,C_A,1}(\alpha_j) \setminus \{\beta_{j,1}\}.$$

The first equalities immediately show that there can be at most one such vertex, say $y_1 = y$ (i.e., $\ell = 1$), and so we have a vertex set S such that:

$$\emptyset \neq S \subset (C_A \setminus \{\beta_{j,1}\}),\ I_{G_A,C_A,1}(\alpha_j) = S \cup \{\beta_{j,1}\} \text{ and } I_{G_A,C_A,1}(y) = S$$

(actually, $S \subset (C_A \cap V)$). Then, since G is 1-twin-free, there is a vertex $z \in V$ which 1-separates α_j and y. If in C_A we replace $\beta_{j,1}$ by z, and with $C_A \cap V_j^* = \{\delta_j, \lambda_j\}$, we obtain a code which is still 1-identifying and optimal in G_A, and does not contain $\beta_{j,1}$. This however contradicts the fact that C_A minimizes $|C_A \cap V^*|$.

So we have proved that $\beta_{j,1} \notin C_A$, for all $j \in J_k$. Then no codeword in V_A^* interferes with G, and $C_A \cap V$ is a 1-identifying code in G.

(b) Conversely, assume that $i_1(G) = i_1(G_A) - 2k$. Let C_A be an optimal 1-identifying code in G_A; then again, $A \subset (C_A \cap V)$, $|C_A \cap V| \leq |C_A| - 2k = i_1(G_A) - 2k = i_1(G)$, and we can assume that $\beta_{j,1} \notin C_A$ for $j \in J_k$. All this implies that

- $C_A \cap V$ is a 1-identifying code in G,
- $C_A \cap V$ has size at most $i_1(G)$, i.e., $C_A \cap V$ is optimal in G,
- $C_A \cap V$ contains A. □

Corollary 4. *Let $r \geq 1$ be an integer, G be an r-twin-free graph containing a subset of vertices A of size k, and G^r be the r-th power of G. We construct the graph $(G^r)_A$ in the same way as in the previous lemma for G.*

Then A is included in at least one optimal r-identifying code in G if and only if $i_1(G^r) = i_1((G^r)_A) - 2k$.

Proof. Follows immediately from Lemmas 2 and 3. □

Thus, the characterization of a subset of vertices included in at least one optimal r-identifying code is obtained through the comparison of two 1-identification numbers. This will be used in the proof of Lemma 8(2), which in turn will help to prove Corollary 21(c). Still, this result is insufficient for our purpose. Unfortunately, it seems difficult to improve it and obtain a similar characterization implicating $i_r(G)$, for $r \geq 2$, because in a construction of the type used for Lemma 3, there are, for $r \geq 2$, greater interferences between the vertices of the graph G we start from, and the vertices we add to get the new graph G_A. See Corollary 21 and the paragraph preceding it.

This is why we are going to give now some more notation and results about identifying codes, which will partially help us to overcome this difficulty in Section 2; the most important tool will be Lemma 6, used for Propositions 16, 17 and 23.

Let $G = (V, E)$ be an r-twin-free graph with n vertices, let $W = \{\{u, v\} : u \in V, v \in V, u \neq v\}$, let $L \subseteq W$ be a *list of pairs* and $M \subseteq V$ be a *list of vertices*. For any code $C \subseteq V$, we denote by $\ell_r(C, L)$ the number of pairs in L which are not r-separated by C, and by $m_r(C, M)$ the number of vertices in M which are not r-covered by C. We say that C is *(r, L, M)-identifying* in G if $\ell_r(C, L) = m_r(C, M) = 0$, and we define the function $\omega_r(L, M)$ as follows:

$$\omega_r(L, M) = \min\{|C| : C \subseteq V, C \text{ is an } (r, L, M)\text{-identifying code in } G\}.$$

Note that, because G is r-twin-free, whatever the sets $L \subseteq W$ and $M \subseteq V$ are, an (r, L, M)-identifying code exists in G ($C = V$ will always do). A code C which is (r, L, M)-identifying in G is said to be *optimal* if $\omega_r(L, M) = |C|$. Of course, if $L = W$ and $M = V$, we have the usual definition of an (optimal) r-identifying code, and $\omega_r(W, V) = i_r(G)$. If $L = M = \emptyset$, then $\omega_r(L, M) = 0$, and conversely.

Let $X \subseteq V$. For any set of pairs $L \subseteq W$ and any set of vertices $M \subseteq V$, we let $L(X)$ be the set of pairs in L which are not r-separated by X, and $M(X)$ be the set of vertices in M not r-covered by X; note that $M(X) = M \setminus (B_r(X) \cap M)$. The important particular case when $L = W$ and $M = V$ yields the notation $W(X)$ and $V(X)$.

Lemma 5. *With the above notation,*

(a) if C is an (r, L, M)-identifying code in G containing X, then $C \setminus X$ is $(r, L(X), M(X))$-identifying in G;

(b) if C^* is an $(r, L(X), M(X))$-identifying code containing X, then C^* is also (r, L, M)-identifying; if C^* is an $(r, L(X), M(X))$-identifying code not containing X, then $C^* \cup X$ is (r, L, M)-identifying.

Proof. (a) All the pairs in L are r-separated by C, and all the vertices in M are r-covered by C. In particular, all the pairs in $L(X)$ are r-separated by C, all the vertices in $M(X)$ are r-covered by C, but these tasks are not performed by X, so $C \setminus X$ must do it.

(b) The code C^* r-separates all the pairs in $L(X)$ and r-covers all the vertices in $M(X)$. The set X r-separates all the pairs in $L \setminus L(X)$ and r-covers all the vertices in $M \setminus M(X)$. This is sufficient to prove the last two assertions of the lemma. □

Lemma 6. *With the above notation, a set X is included in at least one optimal (r, L, M)-identifying code in G if and only if*

$$\omega_r(L, M) = \omega_r(L(X), M(X)) + |X|.$$

In particular, X is included in at least one optimal r-identifying code in G if and only if $i_r(G) = \omega_r(W(X), V(X)) + |X|$.

Proof. (a) Assume that X is included in an optimal (r, L, M)-identifying code C in G: we have $|C| = \omega_r(L, M)$. By the previous lemma, $C_X = C \setminus X$ is $(r, L(X), M(X))$-identifying, and so $\omega_r(L(X), M(X)) \leq |C_X| = \omega_r(L, M) - |X|$.

On the other hand, let C^* be an optimal $(r, L(X), M(X))$-identifying code, and assume that $|C^*| \leq \omega_r(L, M) - |X| - 1$. By the previous lemma, if $X \subseteq C^*$, then C^* is also (r, L, M)-identifying, and if X is not a subset of C^*, then it is $C^* \cup X$ which is (r, L, M)-identifying. In both cases, we obtain an (r, L, M)-identifying code with cardinality less than $\omega_r(L, M)$, which is impossible. So $\omega_r(L(X), M(X)) > \omega_r(L, M) - |X| - 1$, and finally $\omega_r(L(X), M(X)) = \omega_r(L, M) - |X|$.

(b) Assume that X is a set of vertices such that $\omega_r(L(X), M(X)) = \omega_r(L, M) - |X|$, and consider an optimal $(r, L(X), M(X))$-identifying code, C^*. If $X \subseteq C^*$, then C^* is also (r, L, M)-identifying by Lemma 5, which contradicts the previous equality. So X is not included in C^*, and $C = C^* \cup X$ is (r, L, M)-identifying, contains X and its size is at most (actually, is equal to)

$$\omega_r(L(X), M(X)) + |X| = \omega_r(L, M),$$

i.e., C is optimal. □

So we are now able to characterize the inclusion of a set of vertices in an optimal (r, L, M)-identifying code, by handling lists of pairs and of vertices, and by comparing two values of a function which is very similar to the identification number of the graph (see Proposition 12 and its Corollary 13, Propositions 14 and 15, Propositions 18, 19 and their Corollary 20).

2. Complexity results for identifying codes

2.1. Presentation of the problems

We present seven problems dealing with identifying codes.

1) Problem IdR (Identifying Radius):
Instance: A graph $G = (V, E)$ and a code $C \subseteq V$.
Question: What are the nonnegative integers r, if any, such that C is an r-identifying code in G?

Note that, unlike for dominating codes, an r-identifying code is not necessarily $(r+1)$-identifying. An immediate example is $P_3 = x_1x_2x_3$, for which $\{x_1, x_3\}$ is 1-identifying and not 2-identifying (and P_3 is not even 2-identifiable). See also, e.g., [14] for a code which is 1-identifying and not 2-identifying in F_2^5, the binary vector space of dimension five, and [4] for more examples in F_2^n. This is why it would be less interesting to state the previous problem with the question: "What is the smallest nonnegative integer r, if any, such that ..." The following six problems are stated for a fixed integer r, $r \geq 1$; their names are indexed by r.

2) Problem IdC_r (r-Identifying Code with bounded size):
Instance: An r-twin-free graph G and an integer k.
Question: Does G admit an r-identifying code of size at most k?

3) Problem IdN_r (r-Identification Number):
Instance: An r-twin-free graph G.
Output: The r-identification number of G, $i_r(G)$.

4) Problem OIdCS_r (Search for an Optimal r-Identifying Code):
Instance: An r-twin-free graph G.
Search: Determine an optimal r-identifying code in G.

5) Problem Sub-OIdCE_r (Existence of an Optimal r-Identifying Code containing a given Subset):
Instance: An r-twin-free graph $G = (V, E)$ and a nonempty subset of vertices $X \subseteq V$.
Question: Does G admit an optimal r-identifying code containing X?

6) Problem Sub-OIdCS_r (Search for an Optimal r-Identifying Code containing a given Subset):
Instance: An r-twin-free graph $G = (V, E)$ and a nonempty subset of vertices $X \subseteq V$.
Search: Determine, when it exists, an optimal r-identifying code in G containing X.

An algorithm solving Sub-OIdCS_r outputs a suitable code if there is one, or states that no such code exists.

7) Problem Sub-SmIdCS_r (Search for a Smallest r-Identifying Code containing a given Subset):
Instance: An r-twin-free graph $G = (V, E)$ and a nonempty subset of vertices $X \subseteq V$.
Search: Determine an r-identifying code in G containing X, with the smallest size.

Note that, since G is r-twin-free, such codes do exist.

To these problems, we add three subsidiary problems, originating from Subsection 1.3. We recall that, for a graph $G = (V, E)$, we have set $W = \{\{u, v\} : u \in V, v \in V, u \neq v\}$.

A) Problem List-IdC_r ((r, L, M)-Identifying Code with bounded size):
Instance: An r-twin-free graph $G = (V, E)$, a list $L \subseteq W$ of pairs of vertices, a list $M \subseteq V$ of vertices, and an integer k.
Question: Does G admit an (r, L, M)-identifying code of size at most k?

B) Problem List-IdN_r ((r, L, M)-Identification Number):
Instance: An r-twin-free graph $G = (V, E)$, a list $L \subseteq W$ of pairs of vertices, and a list $M \subseteq V$ of vertices.
Output: The minimum size of an (r, L, M)-identifying code in G.

C) Problem List-Sub-OIdCE_r (Existence of an Optimal (r, L, M)-Identifying Code containing a given Subset):
Instance: An r-twin-free graph $G = (V, E)$, a list $L \subseteq W$ of pairs of vertices, a list $M \subseteq V$ of vertices, and a nonempty subset of vertices $X \subseteq V$.
Question: Does G admit an optimal (r, L, M)-identifying code containing X?

We shall give complexity results on these problems mostly insofar as they are useful for the first seven problems.

Remark 7. We can see immediately that List-IdC_r, List-IdN_r and List-Sub-OIdCE_r are at least as difficult as IdC_r, IdN_r and Sub-OIdCE_r, respectively, as the latter problems are subproblems of the former ones, with $L = W$ and $M = V$.

2.2. Known results and motivations

So far, most papers devoted to complexity issues for identifying codes have considered the decision problem IdC_r.

It is easy to see that the problem IdR is polynomial, cf. Corollary 10 below.

The problem IdC_r, as for it, is *NP*-complete for all $r \geq 1$: the case $r = 1$ comes from [6] (see also [8] for a simpler proof) and is stated in Proposition 11, and the case $r > 1$ is from [5], see Proposition 12. In, e.g., [1–3,7–10,23], one can find, in particular, polynomiality or *NP*-completeness results for this problem when restricted to some subclasses of graphs, such as trees, planar graphs, bipartite graphs, interval graphs, permutation graphs or line graphs.

See also [15,16] for the complexity of identification in the binary hypercube.

When dealing with complexity issues, one quite naturally considers the optimization problem, which is here: how to find an optimal r-identifying code? (OIdCS_r). Then one goes to the associated decision problem, IdC_r; once it is proved to be *NP*-complete, we can deduce that OIdCS_r is *NP*-hard. This however does not give an upper bound on the complexity of OIdCS_r, but merely a lower bound. We try to get a better location of OIdCS_r: see Proposition 23(i) which states that OIdCS_r belongs to FP^{NP}. We also feel that results about the respective difficulties of related problems, such as determining the identification number of a graph or finding an optimal identifying code containing a given subset, can be of interest and help to gain a better insight into these issues. Moreover, most of the works cited above deal with the case $r = 1$, whereas we try here, as much as possible, to obtain results that are valid for all $r \geq 1$.

2.3. The results

We want to locate the problems stated in Subsection 2.1 inside the polynomial hierarchy of problems. For the general theory of completeness and hardness in the polynomial hierarchy, we refer to [11]; see also [13] for a comprehensive survey of the main complexity classes, as well as [19] and [22]. From a practical viewpoint, we do not know of polynomial algorithms solving exactly a problem known to be *NP*-hard (and such algorithms simply do not exist if $P \neq NP$): the time required can grow exponentially with the size of the instance (here, the size of the instance is polynomially linked to n, the order of the graph).

Before we try to locate these problems inside the polynomial hierarchy, we can already state a few more results about their respective compared complexities; here, the meaning of "at least as difficult as" is the following: a problem π_1 is at least as difficult as a problem π_2 if an algorithm solving π_1 provides an algorithm for solving π_2 with the same qualitative complexity.

Lemma 8. *Let $r \geq 1$ be an integer.*

(1) The problem IdN_r is at least as difficult as IdC_r.
(2) The problem IdN_1 is at least as difficult as Sub-OIdCE_r.
(3) The problem OIdCS_r is at least as difficult as IdN_r.
(4) The problem Sub-OIdCS_r is at least as difficult as Sub-OIdCE_r.
(5) The problem Sub-SmIdCS_r is at least as difficult as IdN_r, even in the case when X is a singleton.

Proof. (1) With only one call to any algorithm providing $i_r(G)$, the answer to IdN_r, we can give the answer to IdC_r, by comparing $i_r(G)$ and the integer k in the instance of IdC_r. So IdN_r is at least as difficult as IdC_r.

(2) Consider an instance $(G, X \subseteq V)$ of Sub-OIdCE_r. By Corollary 4, and using the notation of Lemma 3, it is sufficient to compute and compare $i_1(G^r)$ and $i_1((G^r)_X) - 2|X|$: the answer to Sub-OIdCE_r is "yes" if and only if equality holds. Now this can be done by using twice an algorithm solving the problem IdN_1, together with negligible operations such as constructing the auxiliary graphs.

The statements (3) and (4) are obvious.

(5) Consider an algorithm solving Sub-SmIdCS_r and run it separately n times, each time with a different singleton $X = \{x\} \subset V$. The smallest code thus obtained gives the r-identification number of G.

Therefore, Sub-SmIdCS_r, with $X = \{x\}$, is at least as difficult as IdN_r. □

We start with easy or already known results. In particular, we give the following lemma without proof.

Lemma 9. *Given an integer $r \geq 1$ and a graph $G = (V, E)$, checking that a given code $C \subseteq V$ is r-identifying is polynomial in the order of the graph.*

Corollary 10. *The problem IdR is polynomial.*

Proof. Here, all we have to do in order to solve IdR is to check whether C is r-identifying, for $r = 0$, $r = 1, \ldots,$ and the number of these checkings is equal to $|V|$. □

Proposition 11. *(See [6,8].) The decision problem IdC_1 is NP-complete.*

The earliest proof is in [6], but the simplest is in [8]. Once the membership of IdC_1 to *NP*, the class of nondeterministic polynomial problems, is established, the *NP*-completeness gives a sort of lower bound on its complexity: the problem IdC_1 is at least as difficult as well-known difficult problems, such as "3-Satisfiability", "3-Dimensional Matching", "Hamiltonian Circuit" or "Partition", and more generally, at least as difficult as any problem in *NP*. Still, *NP*-completeness results are conditional in some sense; if for example $P = NP$, they would lose their interest.

Proposition 12. *(See [5].) Let $r \geq 2$ be an integer. The decision problem IdC_r is NP-complete.*

Corollary 13. *Let $r \geq 1$ be an integer. The decision problem List-IdC_r is NP-complete.*

Proof. First, this problem is in *NP*, since checking a guessed solution can be done in polynomial time (cf. also Lemma 9). Second, it has, as a subproblem, the *NP*-complete problem IdC_r, cf. Remark 7. □

To go further, we need the following notation and additional notions of complexity (see, e.g., [19] or [22]).

The class P^{NP} (also known as Δ_2 in the polynomial hierarchy) contains the decision problems which can be solved by applying, with a number of calls which is polynomial with respect to the size of the instance, a subprogram able to solve an appropriate problem in *NP* (usually, an *NP*-complete problem). The class L^{NP} (also known as Θ_2 and $P_{||}^{NP}$) contains the decision problems which can be solved by applying, with a number of calls which is logarithmic with respect to the size of the instance, a subprogram able to solve an appropriate problem in *NP*. For problems which are not decision problems, these classes are generalized, using "*F*" (for "function") in front of their names; thus, the class FP^{NP} (respectively, FL^{NP}) contains the problems which can be solved by applying, with a number of calls which is polynomial (respectively, logarithmic) with respect to the size of the instance, a subprogram able to solve an appropriate problem in *NP*. Membership to *NP*, L^{NP}, P^{NP}, FL^{NP} or FP^{NP} gives an upper bound on the complexity of a problem (this problem is not more difficult than ...), whereas a hardness result gives a lower bound (this problem is at least as difficult as ...).

The next proposition is easy and uses a very standard argument, see for instance [17].

Proposition 14. *For $r \geq 1$, the problem IdN_r belongs to the class FL^{NP}.*

Proof. Let $\mathcal{A}_r$ be an algorithm which solves the decision problem IdC_r: for any instance (G, k) of IdC_r, it says whether there is an r-identifying code of size k or less in G. This algorithm can be used to solve IdN_r with a number of calls bounded from above by a logarithm in the size of the instance. If n is the order of G, for the instance $(G, k = n)$ of IdC_r, the answer is "yes". Thanks to the standard dichotomous process starting from this initial value, we may compute the size of an optimal r-identifying code in G with at most $\lceil \log n \rceil$ calls to $\mathcal{A}_r$. Since IdC_r is in *NP* (it is actually *NP*-complete, see Proposition 12), we can conclude that IdN_r belongs to FL^{NP}. □

Proposition 15. *For $r \geq 1$, the problem List-IdN_r belongs to the class FL^{NP}.*

Proof. Same argument as in the previous proof, with the problem List-IdC_r, which is in *NP* and is even *NP*-complete (Corollary 13). □

We can locate precisely Sub-OIdCE_r in the hierarchy: we shall prove its membership to L^{NP}, then its L^{NP}-completeness.

Proposition 16. *For $r \geq 1$, the problem Sub-$OIdCE_r$ belongs to the class L^{NP}.*

Proof. Consider an instance $(G, X \subseteq V)$ of Sub-OIdCE_r. By Lemma 6, it is sufficient to compute $i_r(G) = \omega_r(W, V)$ and $\omega_r(W(X), V(X)) + |X|$, and see if $\omega_r(W, V) = \omega_r(W(X), V(X)) + |X|$: the answer to Sub-$\text{OIdCE}_r$ is positive if and only if this equality holds. Now this can be done by using twice an algorithm solving List-IdN_r, together with negligible operations. In turn, as we have just seen, solving List-IdN_r can be done with a logarithmic number of calls to an algorithm solving List-IdC_r, which is in *NP* (Corollary 13).

Alternatively, one can use Lemma 8(2), together with solving IdN_1 by calling a logarithmic number of times an algorithm solving IdC_1. □

Proposition 17. *For $r \geq 1$, the problem List-Sub-$OIdCE_r$ belongs to the class L^{NP}.*

Proof. Same argument as in the previous proof: by Lemma 6, it is sufficient to see if $\omega_r(L, M) = \omega_r(L(X), M(X)) + |X|$ or not. □

Proposition 18. *For $r \geq 1$, the problem Sub-$OIdCE_r$ is L^{NP}-complete.*

Proof. The membership to L^{NP} having just been established, we use the following polynomial reductions:

(i) from the L^{NP}-complete problem Vertex Cover Member [12, Cor. 4.13] to Sub-OIdCE_1 with $X = \{x\}$,

(ii) from Vertex Cover Member to Sub-OIdCE_r with $X = \{x\}$ (for $r \geq 2$), and finally

(iii) from Sub-OIdCE_r with $X = \{x\}$ to Sub-OIdCE_r.

Problem VCM (Vertex Cover Member):
Instance: A graph $G = (V, E)$ and a vertex $x \in V$.
Question: Does G admit an optimal vertex cover containing x?

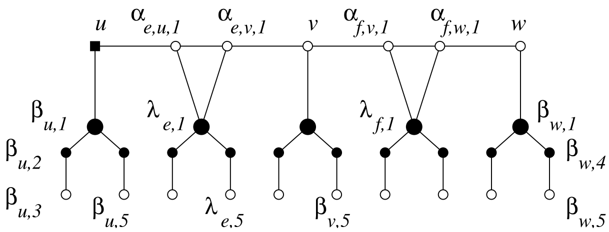

Fig. 2. For $r=1$, construction of V^+ and E^+, starting from the edges $e=uv\in E$ and $f=vw\in E$. Large black circles must belong to any 1-identifying code in G^+. Smaller black circles belong to the 1-identifying code C defined in (3). The vertex u is represented by a black square, because, in the proof of Proposition 18, we assume that it belongs to V^*, hence to C.

A *vertex cover* is a subset $V^*\subseteq V$ such that for each edge $uv\in E$, at least one of u and v belongs to V^*. We shall explain in detail the first reduction, then give a sketch of the proof for the second reduction, and the third one is straightforward.

(i) The polynomial reduction from VCM to Sub-OIdCE$_1$ with $X=\{x\}$ is the following: if $(G=(V,E),x)$ is an instance of VCM, we take as an instance for Sub-OIdCE$_1$ the vertex $x^+=x$ and the graph $G^+=(V^+,E^+)$ constructed as follows, see Fig. 2: for each vertex $v\in V$, we construct

$$V_v^+=\{v,\beta_{v,i}:1\le i\le 5\},\ E_v^+=\{v\beta_{v,1},\beta_{v,1}\beta_{v,2},\beta_{v,2}\beta_{v,3},\beta_{v,1}\beta_{v,4},\beta_{v,4}\beta_{v,5}\};$$

for each edge $e=uv\in E$, we construct

$$V_e^+=\{\alpha_{e,u,1},\alpha_{e,v,1},\lambda_{e,i}:1\le i\le 5\},\ E_e^+=\{u\alpha_{e,u,1},\alpha_{e,u,1}\alpha_{e,v,1},\alpha_{e,v,1}v,$$
$$\alpha_{e,u,1}\lambda_{e,1},\alpha_{e,v,1}\lambda_{e,1},\lambda_{e,1}\lambda_{e,2},\lambda_{e,2}\lambda_{e,3},\lambda_{e,1}\lambda_{e,4},\lambda_{e,4}\lambda_{e,5}\}.$$

The third subscript for some of the vertices in V_e^+ is not necessary but foreshadows the generalization of the construction to any $r\ge 2$. Then G^+ consists of the union of these vertex sets and edge sets. We can see immediately that if C is a 1-identifying code in G^+, then

(a) for every vertex $v\in V$, $\beta_{v,1}\in C$, because $\beta_{v,1}$ is the only vertex that 1-separates $\beta_{v,2}$ and $\beta_{v,3}$, and, for a similar reason, for every edge $e\in E$, $\lambda_{e,1}\in C$;

(b) for every vertex $v\in V$, at least one of the two vertices $\beta_{v,2},\beta_{v,3}$ belongs to C, because $\beta_{v,3}$ must be 1-covered by some codeword. The same is true for $\beta_{v,4}$ and $\beta_{v,5}$, and, similarly, for every edge $e\in E$, for $\lambda_{e,2}$ and $\lambda_{e,3}$, and for $\lambda_{e,4}$ and $\lambda_{e,5}$. As a consequence,

$$|C|\ge|C\cap V|+3(|E|+|V|);\tag{2}$$

(c) for every edge $e=uv\in E$, at least one of the two vertices u and v belongs to C, because the only two vertices that 1-separate $\alpha_{e,u,1}$ and $\alpha_{e,v,1}$ are u and v.

Now we assume that VCM admits a (not necessarily optimal) vertex cover V^*, containing x, in G. Then

$$C=V^*\cup\{\beta_{v,1},\beta_{v,2},\beta_{v,4}:v\in V\}\cup\{\lambda_{e,1},\lambda_{e,2},\lambda_{e,4}:e\in E\}\tag{3}$$

contains x and is 1-identifying in G^+. To prove this, we give below the sets $I_{G^+,C,1}(y)$ of the vertices y associated to the edge $e=uv$, assuming first that $u\in V^*$ and $v\notin V^*$:

$$u:\{u,\beta_{u,1}\},\ \beta_{u,1}:\{u,\beta_{u,1},\beta_{u,2},\beta_{u,4}\},\ \beta_{u,2}:\{\beta_{u,1},\beta_{u,2}\},\ \beta_{u,3}:\{\beta_{u,2}\},$$
$$\beta_{u,4}:\{\beta_{u,1},\beta_{u,4}\},\ \beta_{u,5}:\{\beta_{u,4}\},\ \alpha_{e,u,1}:\{u,\lambda_{e,1}\},\ \alpha_{e,v,1}:\{\lambda_{e,1}\},$$
$$\lambda_{e,1}:\{\lambda_{e,1},\lambda_{e,2},\lambda_{e,3}\},\ v:\{\beta_{v,1}\},\ \beta_{v,1}:\{\beta_{v,1},\beta_{v,2},\beta_{v,4}\};$$

for $i\in\{2,3,4,5\}$, the vertices $\lambda_{e,i}$ and $\beta_{v,i}$ behave exactly like the vertices $\beta_{u,i}$.

Now all these sets are nonempty and distinct, and distinct from the sets $I_{G^+,C,1}(z)$ of vertices z associated to other edges of G. If $u\notin V^*$ and $v\in V^*$, the situation is the same, by symmetry. Finally, if $u\in V^*$ and $v\in V^*$, the conclusion comes from Lemma 1.

Conversely, if C is a (not necessarily optimal) 1-identifying code in G^+, containing $x^+=x$, then, by our preliminary remark (c), $C\cap V$ is a vertex cover in G, which contains x.

If V^* is an optimal vertex cover in G, then the code C defined by (3) is an optimal 1-identifying code in G^+; if not, there would be a 1-identifying code in G^+, say C^+, with $|C^+|<|C|$. But then $C^+\cap V$ would be a vertex cover in G, and inequality (2) would lead to $|C^+\cap V|\le|C^+|-3(|E|+|V|)<|C|-3(|E|+|V|)=|V^*|$, a contradiction.

If C is an optimal 1-identifying code in G^+, then $V^*=C\cap V$ is a vertex cover in G; by (2), it has size $|V^*|\le|C|-3(|E|+|V|)$, and it is optimal: if not, there would be a vertex cover V^+ with $|V^+|<|C|-3(|E|+|V|)$ and the code constructed with V^+ in (3) would be 1-identifying and have fewer elements than C, again a contradiction.

This closes the case $r=1$.

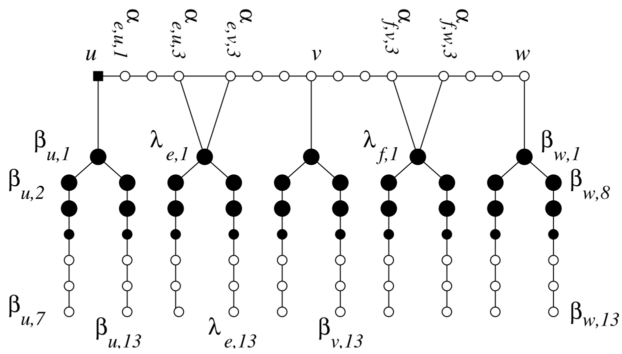

Fig. 3. For $r = 3$, construction of V^+ and E^+. Large black circles must belong to any 3-identifying code in G^+. Smaller black circles belong to the 3-identifying code C defined in (4). We assume that $u \in C$.

(ii) For $r \geq 2$, the graph G^+ constructed for Sub-OIdCE$_r$ with $X = \{x\}$ is a generalization of the previous construction, see Fig. 3 for $r = 3$, where we have more vertices between u and v, and we lengthen the branches growing from the vertices $\beta_{v,1}$ and $\lambda_{e,1}$. More specifically, for each vertex $v \in V$, we construct

$$V_v^+ = \{v, \beta_{v,i} : 1 \leq i \leq 4r+1\},$$

$$E_v^+ = \{v\beta_{v,1}, \beta_{v,1}\beta_{v,2}, \ldots, \beta_{v,2r}\beta_{v,2r+1}, \beta_{v,1}\beta_{v,2r+2}, \ldots, \beta_{v,4r}\beta_{v,4r+1}\};$$

for each edge $e = uv \in E$, we construct

$$V_e^+ = \{\alpha_{e,u,i}, \alpha_{e,v,i} : 1 \leq i \leq r\} \cup \{\lambda_{e,i} : 1 \leq i \leq 4r+1\},$$

$$E_e^+ = \{u\alpha_{e,u,1}, \alpha_{e,u,1}\alpha_{e,u,2}, \ldots, \alpha_{e,u,r-1}\alpha_{e,u,r}, \alpha_{e,u,r}\alpha_{e,v,r}, \alpha_{e,v,r}\alpha_{e,v,r-1}, \ldots,$$

$$\alpha_{e,v,2}\alpha_{e,v,1}, \alpha_{e,v,1}v, \alpha_{e,u,r}\lambda_{e,1}, \alpha_{e,v,r}\lambda_{e,1}, \lambda_{e,1}\lambda_{e,2}, \ldots, \lambda_{e,2r}\lambda_{e,2r+1},$$

$$\lambda_{e,1}\lambda_{e,2r+2}, \ldots, \lambda_{e,4r}\lambda_{e,4r+1}\}.$$

Then again we can make some remarks on an r-identifying code C in G^+:

(a) for every $v \in V$ and $i \in \{1, \ldots, r\}$, $\beta_{v,i} \in C$, because it is the only vertex that r-separates $\beta_{v,i+r}$ and $\beta_{v,i+r+1}$; the same is true for $\beta_{v,i}$, $2r+2 \leq i \leq 3r$. Similarly, for every edge $e \in E$, $\lambda_{e,i} \in C$ for $i \in \{1, \ldots, r\} \cup \{2r+2, \ldots, 3r\}$;

(b) for every $v \in V$, at least one of the $r+1$ vertices $\beta_{v,r+1}, \ldots, \beta_{v,2r+1}$ is a codeword, because $\beta_{v,2r+1}$ is r-covered by C. The same is true for $\beta_{v,3r+1}, \ldots, \beta_{v,4r+1}$, and, for every edge $e \in E$, for $\lambda_{e,r+1}, \ldots, \lambda_{e,2r+1}$, and for $\lambda_{e,3r+1}, \ldots, \lambda_{e,4r+1}$. As a consequence, $|C| \geq |C \cap V| + (2r+1)(|V| + |E|)$;

(c) for every edge $e = uv \in E$, at least one of the two vertices u and v belongs to C, because the only two vertices that r-separate $\alpha_{e,u,r}$ and $\alpha_{e,v,r}$ are u and v.

Assume that VCM admits a vertex cover V^*, containing x, in G. Then it is tedious but straightforward to check that

$$C = V^* \cup \{\beta_{v,i}, \lambda_{e,i} : i \in \{1, \ldots, r+1\} \cup \{2r+2, \ldots, 3r+1\}, v \in V, e \in E\} \tag{4}$$

is r-identifying in G^+ (and contains x). The end of the proof in the general case is then exactly the same as in the case $r = 1$, with the factor 3 replaced by $(2r+1)$.

(iii) Going from Sub-OIdCE$_r$ with $X = \{x\}$ to Sub-OIdCE$_r$ is immediate, noting that the problem with $X = \{x\}$ is a subproblem, whereas Sub-OIdCE$_r$ remains in L^{NP}. □

As a matter of fact, we have proved a result stronger than Proposition 18.

Proposition 19. *For $r \geq 1$, the decision problem Sub-OIdCE$_r$ remains L^{NP}-complete when X is a singleton.*

Corollary 20. *For $r \geq 1$, the problem List-Sub-OIdCE$_r$ is L^{NP}-complete, even when X is a singleton.*

Proof. By Proposition 17, List-Sub-OIdCE$_r$ belongs to L^{NP}, and (cf. Remark 7) it has the L^{NP}-complete problem Sub-OIdCE$_r$ with $X = \{x\}$ as a subproblem. □

The statement (b) of the following corollary of Proposition 18 is not as strong as could be hoped in comparison with (c), because Lemma 3, and even Corollary 4, which is stated for r-identifying codes, only use 1-identification numbers, not r-identification numbers (see the paragraph following Corollary 4, and Conjecture 24).

Corollary 21. *(a) For $r \geq 1$, the problem List-IdN$_r$ is L^{NP}-hard.*
(b) For $r \geq 1$, the problem IdN$_r$ is NP-hard.
(c) The problem IdN$_1$ is L^{NP}-hard.

Proof. (a) We have seen in the proof of Proposition 16 that Sub-OIdCE$_r$ can be solved by calling twice an algorithm solving List-IdN$_r$, and comparing two values of the function ω_r, so List-IdN$_r$ is at least as difficult as Sub-OIdCE$_r$, which is L^{NP}-complete.
(b) Use Lemma 8(1) and the *NP*-completeness of IdC$_r$ (Proposition 12).
(c) Use Lemma 8(2) together with the L^{NP}-completeness of Sub-OIdCE$_1$. □

We now turn to the three search problems OIdCS$_r$ (determine an optimal r-identifying code), Sub-OIdCS$_r$ (given a subset of vertices X, determine an optimal r-identifying code containing X) and Sub-SmIdCS$_r$ (given a subset of vertices X, determine a smallest r-identifying code containing X). The previous results, together with Lemma 8, immediately imply the following corollary, which gives a lower bound on the complexity of these problems.

Corollary 22. *(a) For $r \geq 1$, the problem OIdCS$_r$ is NP-hard; the problem OIdCS$_1$ is L^{NP}-hard.*
(b) For $r \geq 1$, the problem Sub-OIdCS$_r$ is L^{NP}-hard, even in the case when X is a singleton.
(c) For $r \geq 1$, the problem Sub-SmIdCS$_r$ is NP-hard, even in the case when X is a singleton; the problem Sub-SmIdCS$_1$ is L^{NP}-hard, even in the case when X is a singleton.

Proof. (a) Use Lemma 8(3) and the facts that IdN$_r$ is *NP*-hard and IdN$_1$ is L^{NP}-hard.
(b) Use Lemma 8(4) and the fact that Sub-OIdCE$_r$ is L^{NP}-complete, even when X is a singleton.
(c) Use Lemma 8(5) and the facts that IdN$_r$ is *NP*-hard and IdN$_1$ is L^{NP}-hard. □

Then we show that the complexity of these three problems does not go beyond FP^{NP}.

Proposition 23. *(i) For $r \geq 1$, the problem OIdCS$_r$ belongs to the class FP^{NP}.*
(ii) For $r \geq 1$, the problem Sub-OIdCS$_r$ belongs to the class FP^{NP}.
(iii) For $r \geq 1$, the problem Sub-SmIdCS$_r$ belongs to the class FP^{NP}.

Proof. (i) Let $\mathcal{A}_r$ be an algorithm solving List-Sub-OIdCE$_r$. In particular, $\mathcal{A}_r$ can solve instances of Sub-OIdCE$_r$ for which X is a singleton. In a first stage, we show how to solve OIdCS$_r$ by calling $\mathcal{A}_r$ a polynomial number of times.

Let $G = (V, E)$ be an instance of OIdCS$_r$, with n vertices. We recall that for a vertex $x \in V$, for any set of pairs of vertices $L \subseteq W$ and any set of vertices $M \subseteq V$, we let $L(\{x\})$ be the set of pairs in L which are not r-separated by x, and $M(\{x\})$ be the set of vertices in M not r-covered by x.

In a first step, we run $\mathcal{A}_r$ with $G, L_1 = W, M_1 = V$ and different singletons (= vertices) of V until we get a positive answer, i.e., we find a vertex x_1 belonging to at least one (unknown) optimal r-identifying code in G. We set $L_2 = L_1(\{x_1\})$, $M_2 = M_1(\{x_1\})$. In a second step, we run $\mathcal{A}_r$ with G, L_2, M_2 and different vertices of $V \setminus \{x_1\}$ until we find a vertex x_2 belonging to at least one optimal (r, L_2, M_2)-identifying code. Then we set $L_3 = L_2(\{x_2\})$, $M_3 = M_2(\{x_2\})$. At Step i, we run $\mathcal{A}_r$ with G, L_i, M_i and different vertices of $V \setminus \{x_1, \ldots, x_{i-1}\}$ until we find a vertex x_i belonging to at least one optimal (r, L_i, M_i)-identifying code, and we set $L_{i+1} = L_i(\{x_i\})$, $M_{i+1} = M_i(\{x_i\})$. By Lemma 6, we know that $\omega_r(L_i, M_i) = \omega_r(L_{i-1}, M_{i-1}) - 1$. We come to a stop at Step k, $k \leq n$, when $L_k = M_k = \emptyset$, $\omega_r(L_k, M_k) = 0$, which means that $C = \{x_1, \ldots, x_{k-1}\}$ is (r, W, V)-identifying, i.e., C is r-identifying in G.

Since $i_r(G) = \omega_r(L_1, M_1) = \omega_r(L_k, M_k) + (k-1) = k-1$, we see that C has the right size and thus, it is optimal.

How many times do we need to call $\mathcal{A}_r$? If n is the order of the graph, we have at most n steps, in which we call $\mathcal{A}_r$ a decreasing number of times, starting with at most n calls in the first step, so that we have something like at most $n^2/2$ calls to $\mathcal{A}_r$, plus the handling of the lists of pairs L_i and the lists of vertices M_i. (Note however that, since a vertex which has been tried and rejected because it does not belong to any optimal (r, L_j, M_j)-identifying code for the current lists needs not be tested again in the following steps, the number of calls can be reduced to n, approximately.)

This proves that, by calling the algorithm $\mathcal{A}_r$ a polynomial number of times (polynomial with respect to n), we have designed an algorithm which outputs an optimal r-identifying code in G, i.e., solves OIdCS$_r$. This ends our first stage.

In turn, Sub-OIdCE$_r$ can be solved using a logarithmic number of calls to an algorithm solving List-IdC$_r$ (Proposition 16), which is in *NP*. So, all in all, we can solve OIdCS$_r$ by calling a polynomial number of times an algorithm solving a problem in *NP*. This proves that OIdCS$_r$ belongs to FP^{NP}.

(ii) Now we want to call $\mathcal{A}_r$ in order to solve Sub-OIdCS$_r$. First, we run $\mathcal{A}_r$ with X. If the answer is negative, we know that no optimal r-identifying code contains X, and we stop. We assume now that the answer is positive. Then we proceed as in (i): we choose a first vertex in X which will play the part of x_1, we choose a second vertex in X for x_2, and so on until we have used all the vertices in X. Then we look for a vertex belonging to at least one optimal (r, L_j, M_j)-identifying code for the current lists, and we can go on running the algorithm and conclude exactly as previously.

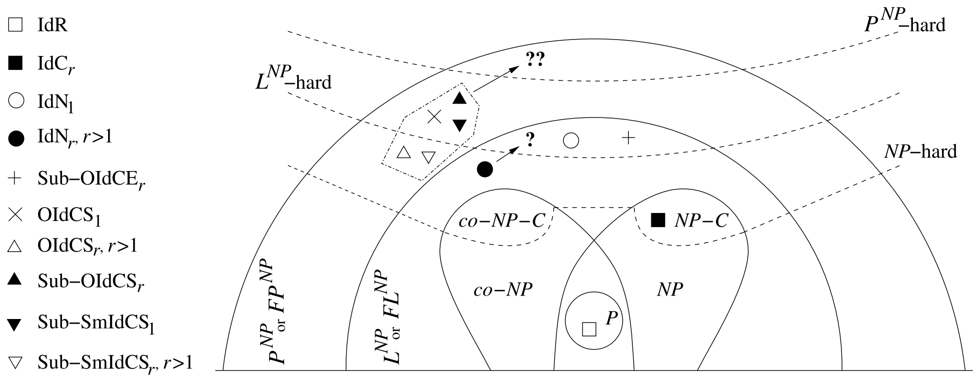

Fig. 4. The locations of our problems in the classes of complexity.

The only difference with (i) is that the first vertices must belong to a specific subset, provided that this subset is contained in an optimal r-identifying code.

(iii) Finally, we want to call $\mathcal{A}_r$ in order to solve Sub-SmIdCS$_r$. We proceed exactly as in (ii), except that we do not need to check whether X is included in an optimal r-identifying code: with $X = \{x_1, \ldots, x_{|X|}\}$ and starting from G, $L_1 = W$ and $M_1 = V$, we construct $L_2 = L_1(\{x_1\})$, $M_2 = M_1(\{x_1\})$, L_3, $M_3, \ldots, L_{|X|+1} = L_{|X|}(\{x_{|X|}\})$, $M_{|X|+1} = M_{|X|}(\{x_{|X|}\})$, without needing to run $\mathcal{A}_r$. Then, once we have used all the vertices in X, we proceed as previously in Cases (i) and (ii), running $\mathcal{A}_r$ with different vertices in $V \setminus X$ until we find one belonging to at least one optimal $(r, L_{|X|+1}, M_{|X|+1})$-identifying code, and so on. At the end, if X belongs to at least one optimal r-identifying code, we have found such a code, and if X does not belong to any optimal r-identifying code, we have found an r-identifying code containing X and with the smallest possible size. □

3. Conclusion

The following results were already known:

- IdR is polynomial (Corollary 10).
- IdC$_r$ is NP-complete for all $r \geq 1$ (Propositions 11 [6,8] and 12 [5]).

We recapitulate below our own results, and present conjectures. For any fixed integer r, $r \geq 1$,

- IdN$_r$ belongs to FL^{NP} (Proposition 14) and is NP-hard (Corollary 21(b)); IdN$_1$ is L^{NP}-hard (Corollary 21(c)).
- Sub-OIdCE$_r$ is L^{NP}-complete (Proposition 18), even if $|X| = 1$ (Proposition 19).
- OIdCS$_r$ belongs to the class FP^{NP} (Proposition 23(i)) and is NP-hard (Corollary 22(a)); OIdCS$_1$ is L^{NP}-hard (Corollary 22(a)).
- Sub-OIdCS$_r$ belongs to the class FP^{NP} (Proposition 23(ii)) and is L^{NP}-hard, even if $|X| = 1$ (Corollary 22(b)).
- Sub-SmIdCS$_r$ belongs to the class FP^{NP} (Proposition 23(iii) and is NP-hard, even if $|X| = 1$ (Corollary 22(c)); Sub-SmIdCS$_1$ is L^{NP}-hard (Corollary 22(c)), even if $|X| = 1$.

These results are represented in Fig. 4, though in a simplified and thus improper way: we make no difference between decision problems and non-decision problems, between P^{NP} and FP^{NP}, ... The four problems IdR, IdC$_r$, IdN$_1$, and Sub-OIdCE$_r$ are located exactly. The problem IdN$_r$ ($r > 1$) is "between" NP-hard and FL^{NP}; in Fig. 4, we place it, for lack of a better knowledge, outside $NP \cup$ co-NP, and below the line L^{NP}-hard, but we conjecture that it is above this line (as is IdN$_1$); this conjecture is represented by an arrow and a question mark in the Figure:

Conjecture 24. *For $r \geq 1$, the problem IdN$_r$ is L^{NP}-hard.*

The two search problems OIdCS$_r$ ($r > 1$) and Sub-SmIdCS$_r$ ($r > 1$) are between NP-hard and FP^{NP}; in Fig. 4, we place them, for lack of a better knowledge, outside FL^{NP}, and below the line L^{NP}-hard, but we conjecture below that they are P^{NP}-hard. The three search problems OIdCS$_1$, Sub-OIdCS$_r$, and Sub-SmIdCS$_1$ are between L^{NP}-hard and FP^{NP}; in Fig. 4, we place them, for lack of a better knowledge, outside FL^{NP}, and below the line P^{NP}-hard. We conjecture that they are also P^{NP}-hard; this multiple conjecture is represented by an arrow and a double question mark in the figure:

Conjecture 25. *For $r \geq 1$, the problems OIdCS$_r$, Sub-OIdCS$_r$ and Sub-SmIdCS$_r$ are P^{NP}-hard, even when X is a singleton.*

Acknowledgements

We wish to thank the two referees for their very careful reading and helpful remarks and suggestions.

References

[1] D. Auger, Identifying codes in trees and planar graphs, Electron. Notes Discrete Math. 34 (2009) 585–588.
[2] D. Auger, Minimal identifying codes in trees and planar graphs with large girth, European J. Combin. 31 (2010) 1372–1384.
[3] D. Auger, I. Charon, O. Hudry, A. Lobstein, Complexity results for identifying codes in planar graphs, Int. Trans. Oper. Res. 17 (2010) 691–710.
[4] I. Charon, G. Cohen, O. Hudry, A. Lobstein, New identifying codes in the binary Hamming space, European J. Combin. 31 (2010) 491–501.
[5] I. Charon, O. Hudry, A. Lobstein, Minimizing the size of an identifying or locating-dominating code in a graph is NP-hard, Theoret. Comput. Sci. 290 (2003) 2109–2120.
[6] G. Cohen, I. Honkala, A. Lobstein, G. Zémor, On identifying codes, in: Proceedings of DIMACS Workshop on Codes and Association Schemes '99, vol. 56, Piscataway, USA, 2001, pp. 97–109.
[7] F. Foucaud, Aspects combinatoires et algorithmiques des codes identifiants dans les graphes, Université de Bordeaux 1, France, December 2012, Thèse de Doctorat, 194 pages (in English).
[8] F. Foucaud, Decision and approximation complexity for identifying codes and locating-dominating sets in restricted graph classes, J. Discrete Algorithms 31 (2015) 48–68.
[9] F. Foucaud, S. Gravier, R. Naserasr, A. Parreau, P. Valicov, Identifying codes in line graphs, J. Graph Theory 73 (2013) 425–448.
[10] F. Foucaud, G. Mertzios, R. Naserasr, A. Parreau, P. Valicov, Identification, location-domination and metric dimension on interval and permutation graphs, II, algorithms and complexity, Algorithmica (2016), in press, available at http://arxiv.org/abs/1405.2424.
[11] M.R. Garey, D.S. Johnson, Computers and Intractability, A Guide to the Theory of NP-Completeness, Freeman, New York, 1979.
[12] E. Hemaspaandra, H. Spakowski, J. Vogel, The complexity of Kemeny elections, Theoret. Comput. Sci. 349 (2005) 382–391.
[13] L. Hemaspaandra, Complexity classes, in: Handbook of Discrete and Combinatorial Mathematics, CRC Press, Boca Raton, 2000, pp. 1085–1090.
[14] I. Honkala, On the identifying radius of codes, in: Proceedings of the 7th Nordic Combinatorial Conference, Turku, Finland, 1999, pp. 39–43.
[15] I. Honkala, A. Lobstein, On identifying codes in binary Hamming spaces, J. Combin. Theory Ser. A 99 (2002) 232–243.
[16] I. Honkala, A. Lobstein, On the complexity of the identification problem in Hamming spaces, Acta Inform. 38 (2002) 839–845.
[17] O. Hudry, On the complexity of Slater's problems, European J. Oper. Res. 203 (2010) 216–221.
[18] O. Hudry, A. Lobstein, More results on the complexity of domination problems in graphs, Int. Trans. Oper. Res. (2016), in press.
[19] D.S. Johnson, A catalog of complexity classes, in: J. van Leeuwen (Ed.), Handbook of Theoretical Computer Science, Vol. A: Algorithms and Complexity, Elsevier, Amsterdam, 1990, pp. 67–161.
[20] M.G. Karpovsky, K. Chakrabarty, L.B. Levitin, On a new class of codes for identifying vertices in graphs, IEEE Trans. Inform. Theory IT-44 (1998) 599–611.
[21] A. Lobstein, Watching systems, identifying, locating-dominating and discriminating codes in graphs, a bibliography, http://www.perso.enst.fr/~lobstein/debutBIBidetlocdom.pdf.
[22] C.H. Papadimitriou, Computational Complexity, Addison–Wesley, Reading, 1994.
[23] A. Parreau, Problèmes d'identification dans les graphes, Thèse de doctorat, Université de Grenoble, France, July 2012, 214 pages.