Unique (Optimal) Solutions: Complexity Results for Identifying and Locating-Dominating Codes

Olivier Hudry, Antoine Lobstein

Unique (Optimal) Solutions: Complexity Results for Identifying and Locating-Dominating Codes

Olivier Hudry
LTCI, Télécom ParisTech, Université Paris-Saclay
46, rue Barrault, 75634 Paris Cedex 13 - France
olivier.hudry@telecom-paristech.fr

Antoine Lobstein
Centre National de la Recherche Scientifique
Laboratoire de Recherche en Informatique, UMR 8623,
Université Paris-Sud, Université Paris-Saclay
Bâtiment 650 Ada Lovelace, 91405 Orsay Cedex - France
antoine.lobstein@lri.fr

September 27, 2018

Abstract

We investigate the complexity of four decision problems dealing with the *uniqueness* of a solution in a graph: "Uniqueness of an r-Locating-Dominating Code with bounded size" (U-LDC$_r$), "Uniqueness of an Optimal r-Locating-Dominating Code" (U-OLDC$_r$), "Uniqueness of an r-Identifying Code with bounded size (U-IdC$_r$), "Uniqueness of an Optimal r-Identifying Code" (U-OIdC$_r$), for any fixed integer $r \geq 1$.

In particular, we describe a polynomial reduction from "Unique Satisfiability of a Boolean formula" (U-SAT) to U-OLDC$_r$, and from U-SAT to U-OIdC$_r$; for U-LDC$_r$ and U-IdC$_r$, we can do even better and prove that their complexity is the same as that of U-SAT, up to polynomials. Consequently, all these problems are *NP*-hard, and U-LDC$_r$ and U-IdC$_r$ belong to the class *DP*.

Key Words: Complexity Theory, Graph Theory, Uniqueness of Solution, Polynomial Reduction, Locating-Dominating Codes, Identifying Codes

1 Introduction

We intend to locate in the classes of complexity some problems dealing with the existence of a *unique* identifying or locating-dominating code in a given graph.

Uniqueness of solutions has been studied in a few papers (see, e.g., [1], [2], [3], [4], [5], [6]) and may be seen as part of the wider and unexplored issue of the number of solutions of a problem.

1.1 Identifying and Locating-Dominating Codes

For graph theory, we refer to, e.g., [7] or [8].

For identification in graphs, see the seminal paper [9]; for locating-dominating codes, see the first papers [10] and [11]. For both, see also the large bibliography at [12], where almost 400 references show that these topics are burgeoning.

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* $P_k = x_1 x_2 \dots x_k$ is a sequence of k distinct vertices x_i, $1 \le i \le k$, such that $x_i x_{i+1}$ is an edge for $i \in \{1, 2, \dots, k-1\}$. The *length* of P_k is its number of edges, $k - 1$. A *cycle* $\mathcal{C}_k = x_1 x_2 \dots x_k$ is a sequence of k distinct vertices x_i, $1 \le i \le k$, where $x_i x_{i+1}$ is an edge for $i \in \{1, 2, \dots, k-1\}$, and $x_k x_1$ is also an edge; its length is k.

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. 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 $B_r(x) = B_r(y)$, $x \neq y$, 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-dominates itself. A set W is said to r-dominate a set Z if every vertex in Z is r-dominated by at least one vertex of W, or equivalently: $Z \subseteq B_r(W)$. When three vertices x, y, z are such that $x \in B_r(z)$ and $y \notin B_r(z)$, 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*.

We say that C is an *r-identifying code* [9] if all the sets $B_r(v) \cap C$, $v \in V$, 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*.

We say that C is an *r-locating-dominating code* (r-LD code for short) [11], [10] if all the sets $B_r(v) \cap C$, $v \in V \setminus C$, are nonempty and distinct: in other words, every vertex is r-dominated by C (since a codeword dominates itself), and every pair of non-codewords is r-separated by C. We denote by $LD_r(G)$ the smallest cardinality of an r-locating-dominating code in G, and call it the *r-location-domination number* of G; any r-LD code C such that $|C| = LD_r(G)$ is said to be *optimal*.

For the needs of Theorems 19 and 34, we give the following obvious characterization: a code C is r-identifying (respectively, r-LD) if and only if (a) for every vertex $x \in V$, $B_r(x) \cap C \neq \emptyset$, and (b) for every pair of distinct vertices $x^i \in V$, $x^j \in V$ (respectively, $x^i \in V \setminus C$, $x^j \in V \setminus C$), we have

$$\big(B_r(x^i) \Delta B_r(x^j)\big) \cap C \neq \emptyset, \tag{1}$$

where Δ stands for the symmetric difference.

Note that, when dealing with locating-dominating codes, we shall rather use the word "dominate", whereas for identifying codes, we shall prefer "cover".

It is known that the following two decision problems, stated for any integer $r \geq 1$, are *NP*-complete (see below Propositions 13 from [10], [13], and 29 from [14], [13]):

Problem LDC$_r$ (r-Locating-Dominating Code with bounded size):
Instance: A graph G and an integer k.
Question: Does G admit an r-locating-dominating code of size at most k?

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?

In this paper, we are interested in the following four problems, which deal with the *uniqueness* of a solution, and we are going to locate them in the classes of complexity.

Problem U-LDC$_r$ (Unique r-Locating-Dominating Code with bounded size):
Instance: A graph G and an integer k.
Question: Does G admit a *unique* r-locating-dominating code of size at most k?

Problem U-OLDC$_r$ (Unique Optimal r-Locating-Dominating Code):
Instance: A graph G.
Question: Does G admit a *unique optimal* r-locating-dominating code?

Problem U-IdC$_r$ (Unique r-Identifying Code with bounded size):
Instance: An r-twin-free graph G and an integer k.
Question: Does G admit a *unique* r-identifying code of size at most k?

Problem U-OIdC$_r$ (Unique Optimal r-Identifying Code):
Instance: An r-twin-free graph G.
Question: Does G admit a *unique optimal* r-identifying code?

Our results are the following: we give polynomial reductions from "Unique Satisfiability of a Boolean formula" (U-SAT) to U-OLDC$_r$, as well as from U-SAT to U-OIdC$_r$; we prove that U-LDC$_r$ and U-IdC$_r$ have the same complexity as U-SAT, up to polynomials. As a consequence, all these problems are *NP*-hard, and U-LDC$_r$ and U-IdC$_r$ belong to the class *DP*. The problems U-OLDC$_r$ and U-OIdC$_r$ belong "only" to the class L^{NP}, which contains *DP*.

In a previous work [15], we have investigated the complexity of the existence of, and of the search for, optimal r-identifying codes, as well as optimal r-identifying codes containing a given subset of vertices; see also [13], [14, Sec. 5]. In a forthcoming work, we extend the present study on uniqueness issues to Boolean satisfiability and graph colouring [16], Vertex Cover and Dominating Set (as well as its generalization to domination within distance r) [17], and Hamiltonian Cycle [18]. At the other end, there has been research on *how many* optimal r-identifying codes can exist in a graph [19], and on the structure of the ensemble of optimal r-locating-dominating codes [20] and of optimal r-identifying codes [21].

For other works in the area of complexity, see, e.g., [22], [23], [24], [25], [26], [27], and [28], which establish, in particular, polynomiality or *NP*-completeness results for the identification problem when restricted to some subclasses of graphs, such as trees, planar graphs, bipartite graphs, interval graphs or line graphs. See also [29], [30] and [31] for approximation issues for both identifying and locating-dominating codes.

In the sequel, we shall also need the following tools, which constitute classical definitions related to Boolean satisfiability.

We consider a set $\mathcal{X}$ of n *Boolean variables* x_i and a set $\mathcal{C}$ of m *clauses*, each clause c_j containing κ_j *literals*, a literal being a variable x_i or its complement $\overline{x}_i$. A *truth assignment* for $\mathcal{X}$ sets the variable x_i to TRUE, also denoted by T, and its complement to FALSE (or F), or *vice-versa.* A truth assignment is said to *satisfy* the clause c_j if c_j contains at least one true literal, and to satisfy the set of clauses $\mathcal{C}$ if every clause contains at least one true literal. The following decision problem, for which the size of the instance is polynomially linked to $n+m$, is a classical problem in complexity.

Problem SAT (Satisfiability):
Instance: A set $\mathcal{X}$ of variables, a collection $\mathcal{C}$ of clauses over $\mathcal{X}$, each clause containing at least two different literals.
Question: Is there a truth assignment for $\mathcal{X}$ that satisfies $\mathcal{C}$?

We shall also need the variant U-SAT of SAT (see [1], [32]), which has the same instance as SAT but the question now is: "Is there a *unique* truth assignment...?".

We shall give in Proposition 2 what we need to know about the complexity of this problem. We now provide the necessary definitions and notation for complexity.

1.2 Necessary Notions in Complexity

We expound here, not too formally, the notions of complexity that will be needed in the sequel. We refer the reader to, e.g., [33], [34], [35] or [36] for more on this topic.

A *decision problem* is of the type "Given an instance I and a property $\mathcal{PR}$ on I, is $\mathcal{PR}$ true for I?", and has only two solutions, "yes" or "no". The class P will denote the set of problems which can be solved by a *polynomial* (time) algorithm, and the class *NP* the set of problems which can be solved by a *nondeterministic polynomial* algorithm. A *polynomial reduction* from a decision problem π_1 to a decision problem π_2 is a polynomial transformation that maps any instance of π_1 into an "equivalent" instance of π_2, that is, an instance of π_2 admitting the same answer as the instance of π_1; in this case, we shall write $\pi_1 \rightarrow_p \pi_2$. Cook [37] proved that there is one problem in *NP*, namely SAT, to which every other problem in *NP* can be polynomially reduced. Thus, in a sense, SAT is the "hardest" problem inside *NP*. Other problems share this property in *NP* and are called *NP-complete* problems; their class is denoted by *NP*-complete or *NP-C*. The way to show that a decision problem π is *NP*-complete is, once it is proved to be in *NP*, to choose some *NP*-complete problem π_1 and to polynomially reduce it to π. From a practical viewpoint, the *NP*-completeness of a problem π implies that we do not know any polynomial algorithm solving π, and that, under the assumption $P \neq NP$, which is widely believed to be true, no such algorithm exists: the time required can grow exponentially with the size of the instance (when the instance is a graph, the size is polynomially linked to its order).

The *complement* of a decision problem, "Given I and $\mathcal{PR}$, is $\mathcal{PR}$ true for I?", is "Given I and $\mathcal{PR}$, is $\mathcal{PR}$ false for I?". The class *co-NP* (respectively, *co-NP*-complete or *co-NP-C*) is the class of the problems which are the complement of a problem in *NP* (respectively, in *NP*-complete).

For problems which are not necessarily decision problems, a *Turing reduction* from a problem π_1 to a problem π_2 is an algorithm $\mathcal{A}$ that solves π_1 using a (hypothetical) subprogram $\mathcal{S}$ solving π_2 such that, if $\mathcal{S}$ were a polynomial algorithm for π_2, then $\mathcal{A}$ would be a polynomial algorithm for π_1. Thus, in this sense, π_2 is "at least as hard" as π_1. A problem π is *NP-hard* (respectively, *co-NP-hard*) if there is a Turing reduction from some *NP*-complete (respectively, co-*NP*-complete) problem to π [34, p. 113].

Remark 1 *Note that with these definitions, NP-hard and co-NP-hard coincide [34, p. 114].*

The notions of completeness and hardness can of course be extended to classes other than *NP* or co-*NP*. *NP*-hardness is defined differently in [38] and [39]: there, a problem π is *NP-hard* if there is a *polynomial* reduction from some *NP*-complete problem to π; this may lead to confusion (see Section 5).

Finally we introduce the classes P^{NP} (also known as Δ_2 in the hierarchy of classes) and L^{NP} (also denoted by $P^{NP[O(\log n)]}$ or Θ_2), which contain the decision 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* (usually, an *NP*-complete problem); and the class *DP* [40] (or DIF^P [1] or BH_2 [35], [41], ...) as the class of languages (or problems) L such that there are two languages $L_1 \in NP$ and $L_2 \in$ co-NP satisfying $L = L_1 \cap L_2$. This class is not to be confused with $NP \cap$ co-NP (see the warning in, e.g., [36, p. 412]); actually, *DP* contains $NP \cup$ co-NP and is contained in L^{NP}. See Figure 1.

Membership to *P*, *NP*, co-*NP*, *DP*, L^{NP} or P^{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 ...). Still, such results are conditional in some sense; if for example *P*=*NP*, they would lose their interest. But it is not known whether or where the classes of complexity collapse.

The problem SAT is one of the most well-known *NP*-complete problems [37], [34, p. 39, p. 46 and p. 259]. The following result is easy.

Proposition 2 *The problem U-SAT is NP-hard (and co-NP-hard by Remark 1), and belongs to the class DP [32].* $\Diamond$

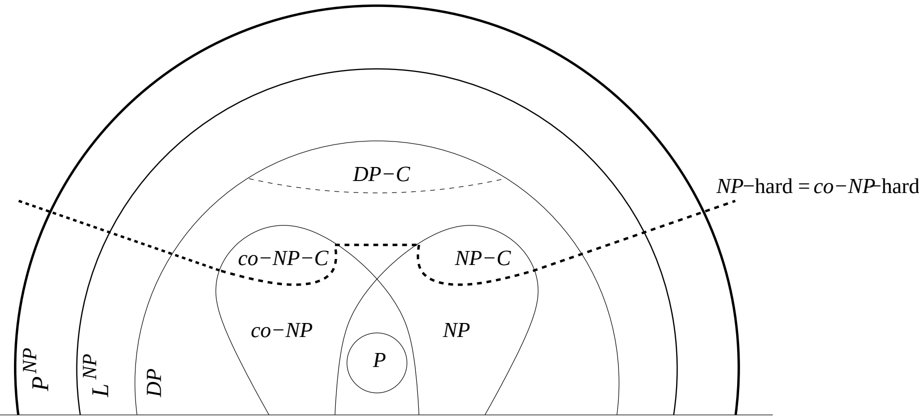

Figure 1: Some classes of complexity.

Remark 3 *It is not known whether U-SAT is DP-complete: in [36, p. 415], it is said that "U-SAT is not believed to be DP-complete". In [1], it is shown that there exists one oracle under which U-SAT is not DP-complete; and one oracle under which it is, if NP≠co-NP.*

We are now ready to investigate the problems of the uniqueness of identifying and LD codes.

2 Some Easy Preliminary Results

These results are as old as the definitions of identifying and LD codes.

Lemma 4 *(a) For any graph $G = (V, E)$ of order n and any integer $r \geq 1$, we have*

$$LD_r(G) \geq \lceil \log_2(n - LD_r(G) + 1) \rceil. \tag{2}$$

(b) For any integer $r \geq 1$ and any r-twin-free graph $G = (V, E)$ of order n, we have

$$i_r(G) \geq \lceil \log_2(n + 1) \rceil. \tag{3}$$

Proof. (a) Let C be any r-LD code in G. All the $n - |C|$ non-codewords $v \in V \setminus C$ must be given nonempty and distinct sets $B_r(v) \cap C$ constructed with the $|C|$ codewords, so $2^{|C|} - 1 \geq n - |C|$, from which (2) follows when C is optimal; (b) the argument is the same, but we have to consider all the n vertices $v \in V$, so $2^{|C|} - 1 \geq n$. $\lozenge$

Lemma 5 *Let $r \geq 2$ be any integer and $G = (V, E)$ be a graph.*

(a) A code C is 1-locating-dominating in G^r, the r-th power of G, if and only if it is r-locating-dominating in G.

(b) A code C is 1-identifying in G^r if and only if it is r-identifying in G.

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

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

so if for all $v \in V \setminus C$, the sets on the left-hand side of the equality are nonempty and distinct, then the sets on the right-side also are, and *vice-versa*; (b) same proof, for all $v \in V$. $\Diamond$

The following obvious lemma is often used implicitly; we give it without proof.

Lemma 6 *Let $r \ge 1$ be any integer and $G = (V, E)$ be a graph.*
(a) If C is r-locating-dominating in G, so is any set $S \supset C$.
(b) If C is r-identifying in G, so is any set $S \supset C$. $\Diamond$

3 Locating-Dominating Codes

After some necessary preliminary results, we are going to prove, for $r \ge 2$ and $q \ge 1$, the following polynomial reductions:
U-SAT $\to_p$ U-LDC$_1$ and U-SAT $\to_p$ U-OLDC$_1$ (Theorem 14),
U-SAT $\to_p$ U-LDC$_r$ and U-SAT $\to_p$ U-OLDC$_r$ (Theorem 15),
U-LDC$_{qr}$ $\to_p$ U-LDC$_q$ and U-OLDC$_{qr}$ $\to_p$ U-OLDC$_q$ (Proposition 18),
U-LDC$_1$ $\to_p$ U-SAT (Theorem 19).
The consequence of these reductions is that, for $r \ge 1$, U-LDC$_r$ and U-OLDC$_r$ are *NP*-hard, and that U-SAT and U-LDC$_r$ have equivalent complexities; as a result, U-LDC$_r$ belongs to *DP*. We shall also show that U-OLDC$_r$ belongs to the class L^{NP} (Proposition 22).

We do not have that U-OLDC$_r$ belongs to *DP* for lack of a polynomial reduction from U-OLDC$_1$ to U-SAT; we conjecture that such a reduction does not exist and that U-OLDC$_r \notin DP$ (see also Conclusion).

Also note that the polynomial reduction U-SAT $\to_p$ U-LDC$_1$ is a consequence of the chain of reductions U-SAT $\to_p$ U-LDC$_r$ $\to_p$ U-LDC$_1$; we still give Theorem 14 and its proof, because it constitutes a preliminary step for the proof of Theorem 15.

3.1 Preliminary Results

Lemma 7 *Let $h \ge 1$ and $r \ge 1$ be integers; let G be a graph of order $2^h - 1 + h$ with $LD_r(G) = h$. Then:*

(a) no vertex r-dominating 2^{h-1}, or fewer, vertices can belong to an optimal r-locating-dominating code in G;

(b) no vertex r-dominating $2^{h-1} + h + 1$, or more, vertices can belong to an optimal r-locating-dominating code in G.

Proof. Let C be any optimal r-LD code in G: $|C| = LD_r(G) = h$. Because there are $2^h - 1$ non-codewords, all the nonempty subsets of C coincide with all the nonempty, distinct sets $B_r(v) \cap C$, $v \in V \setminus C$. Then every codeword c appears exactly 2^{h-1} times in these subsets, which means that c r-dominates exactly 2^{h-1} non-codewords; since it r-dominates between one (itself) and h codewords, all in all it r-dominates between $2^{h-1} + 1$ and $2^{h-1} + h$ vertices, and (a) and (b) follow. $\diamondsuit$

The following lemma is easy, and we prove only its last assertion.

Lemma 8 *(a) The path $P_5 = x_1x_2x_3x_4x_5$ admits only one optimal 1-locating-dominating code, $C = \{x_2, x_4\}$.*

(b) If we construct the graph $G_A = (V_A, E_A)$ by adding to P_5 a vertex denoted by A together with the edge x_2A, then A is 1-dominated by x_2, but x_1 and A are not 1-separated by C.

(c) If G_A is plunged in a larger graph G^+ with only A linked to the outside, then every optimal 1-locating-dominating code C^+ in G^+ contains x_2 and x_4. At most one additional codeword, x_1 or A, may be necessary in $V_A \cap C^+$.

Proof. (c) Let C^+ be any optimal 1-LD code in G^+. (i) If A is a codeword, obviously C^+ contains x_2 and x_4, and no other codeword in P_5. (ii) The same is true if A is not a codeword, but is 1-dominated by at least one outside codeword. (iii) Otherwise, C^+ contains x_2, to 1-dominate A, it contains x_1, otherwise x_1 and A are not 1-separated by C^+, and it contains x_4, which is the only vertex which 1-dominates x_5 and 1-separates A and x_3 at the same time. $\diamondsuit$

The statements of the following lemma have been given, although in a different way, in [13, proof of Lemma 3.1]; for completeness, we give here the (easy) proof.

Lemma 9 *Let $G_i = (V_i, E_i)$ be the following graph: $V_i = \{x_i, \overline{x}_i, a_i, b_i, d_i, f_i, g_i\}$ and $E_i = \{a_ib_i, b_ix_i, b_i\overline{x}_i, x_id_i, \overline{x}_id_i, d_if_i, x_ig_i, \overline{x}_ig_i\}$, and C_i be a 1-locating-dominating code in G_i, see Figure 2. Then:*

(a) at least one of x_i and $\overline{x}_i$ belong to C_i;

(b) at least two more codewords necessarily belong to C_i, so that we have $LD_1(G_i) \geq 3$;

(c) we have $LD_1(G_i) = 3$, and $\{x_i, b_i, d_i\}$ and $\{\overline{x}_i, b_i, d_i\}$ are the only optimal 1-locating-dominating codes in G_i;

(d) if G_i is plunged in a larger graph G^+, with only x_i and $\overline{x}_i$ linked to the outside, then every 1-locating-dominating code in G^+ contains at least three codewords inside V_i.

Proof. (a) Because x_i and $\overline{x}_i$ have the same neighbours in G_i. (b) Because a_i and f_i must be 1-dominated by C_i, we have $|C_i \cap \{a_i, b_i\}| \geq 1$ and $|C_i \cap$

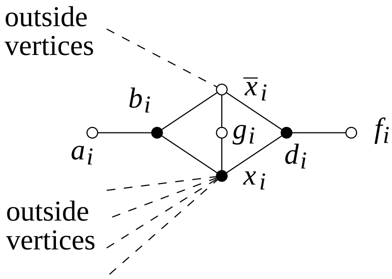

Figure 2: The graph G_i defined in Lemma 9. The black vertices form one of the two optimal 1-LD codes in G_i.

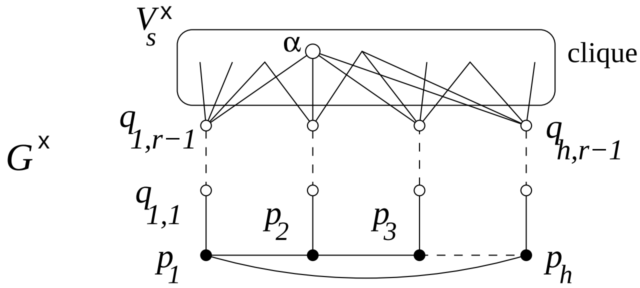

Figure 3: The graph $G^\times$ has a unique optimal r-LD code, $V_p^\times$.

$\{d_i, f_i\}| \geq 1$. Alternatively, use (2) in Lemma 4. (c) Assume that it is x_i that belongs to C_i. Then taking a_i and f_i would not 1-dominate $\overline{x}_i$, and taking a_i and d_i, or b_i and f_i, would not 1-separate $\overline{x}_i$ and f_i, or $\overline{x}_i$ and a_i, respectively; on the other hand, $\{x_i, b_i, d_i\}$ is 1-LD. (d) Only x_i and $\overline{x}_i$ can be 1-dominated by the outside, and a_i, f_i and g_i have to be 1-dominated by the code, so anyway at least three codewords are necessary inside V_i. $\diamondsuit$

The previous two lemmas will be used in the proof of Theorem 14. In particular, Lemma 8(a) gives the example of a graph, P_5, with a unique optimal 1-LD code. We want to have the same for any $r > 1$ (see Proposition 10(a)), in view of Theorem 15. We shall proceed as follows (see Figure 3):

We set $h = 2r + 1$, even if everything that follows also holds for any $h \geq 2r + 1$. Let $G_p^\times = (V_p^\times, E_p^\times)$ be the cycle $\mathcal{C}_h$ of length h, with $V_p^\times = \{p_i : 1 \leq i \leq h\}$. Then we construct $G_q^\times = (V_q^\times, E_q^\times)$, with $V_q^\times = \{q_{i,j} : 1 \leq i \leq h, 1 \leq j \leq r-1\}$ and $E_q^\times = \cup_{1 \leq i \leq h}\{q_{i,j}q_{i,j+1} : 1 \leq j \leq r-2\}$. The set of edges between $G_p^\times$ and $G_q^\times$ is $E_{p,q}^\times = \{p_i q_{i,1} : 1 \leq i \leq h\}$. Next, we construct $G_s^\times = (V_s^\times, E_s^\times)$ with $V_s^\times = \{s_i : 1 \leq i \leq 2^h - 1 - (r-1)h\}$ and $E_s^\times = \{s_{i_1}s_{i_2} : 1 \leq i_1 < i_2 \leq |V_s^\times|\}$, i.e., $G_s^\times$ is a clique.

We set $V^\times = V_p^\times \cup V_q^\times \cup V_s^\times$.

In order to define the set $E_{q,s}^\times$ of edges between $\{q_{i,r-1} : 1 \leq i \leq h\}$

and $V_s^\times$, we introduce, for every vertex $v \in V_q^\times \cup V_s^\times$, the *signature* of v as the set $B_r(v) \cap V_p^\times$ of the elements of the cycle that r-dominate v, and we wish to have nonempty and distinct signatures. Since

(a) the h vertices in $V_p^\times$ can provide $2^h - 1$ such signatures,
(b) $|V_q^\times \cup V_s^\times| = |V_q^\times| + |V_s^\times| = 2^h - 1$,
(c) the vertices in $V_q^\times$ have nonempty and different signatures (in particular, thanks to the fact that $h \geq 2r$, all the vertices $q_{i,1}$ have signatures of size $2r-1$),
(d) a vertex in $V_s^\times$ which is linked (respectively, not linked) to $q_{i,r-1}$ is at distance equal to (respectively, greater than) r from p_i,

we can see that it is possible to construct $E_{q,s}^\times$ in such a way that the vertices in $V_s^\times$ have nonempty signatures which are different inside $V_s^\times$, and different from those for $V_q^\times$. In particular, in $V_s^\times$ there is a vertex which has signature equal to $V_p^\times$; we denote this vertex by α. Note also that we could not have more vertices with this signature property.

We set $E^\times = E_p^\times \cup E_{p,q}^\times \cup E_q^\times \cup E_{q,s}^\times \cup E_s^\times$ and $G^\times = (V^\times, E^\times)$. The order of $G^\times$ is $n^\times = 2^h - 1 + h$.

We claim that, for a fixed $r \geq 2$ and $h = 2r+1$, $C = V_p^\times$ is the *unique* optimal r-LD code in $G^\times$; we shall prove it by going through the following three easy facts.

Fact 1 *For any $r \geq 2$ and $h = 2r+1$, the code $C = V_p^\times$ is an optimal r-locating-dominating code in $G^\times$.*

Proof. When $C = V_p^\times$, the signatures are the sets $B_r(v) \cap C$, for $v \in V^\times \setminus C$. By construction, they are all nonempty and distinct, hence C is r-LD. The optimality comes from (2) in Lemma 4. $\Diamond$

Fact 2 *For any $r \geq 2$ and $h = 2r+1$, the graph $G^\times$ meets the conditions of Lemma 7.*

Proof. Because $LD_r(G^\times) = |V_p^\times| = h$ and $G^\times$ has order $2^h - 1 + h$. $\Diamond$

Fact 3 *For any $r \geq 2$ and $h = 2r+1$, no vertex in $V_q^\times \cup V_s^\times$ can belong to any optimal r-locating-dominating code C in $G^\times$.*

Proof. Because $V_s^\times$ is a clique, every vertex $q_{i,j}$, $1 \leq i \leq h$, $1 \leq j \leq r-1$, is within distance r from every vertex in $V_s^\times$; so every $q_{i,j}$ r-dominates at least $|V_s^\times| = 2^h - 1 - (r-1)h$ vertices. This number is greater than $2^{h-1} + h + 1$ for $r \geq 2$ and $h = 2r+1$. The same is true for the vertices in $V_s^\times$. We can conclude using Lemma 7(b). $\Diamond$

We are now ready to prove the following proposition.

Proposition 10 *Let $r \geq 2$ and $h = 2r+1$. Then:*
(a) the only optimal r-locating-dominating code in $G^\times$ is $C = V_p^\times$;

(b) if $G^\times$ is plunged in a larger graph $G^+ = (V^+, E^+)$, with only α linked to the outside, then every optimal r-locating-dominating code in G^+ contains $V_p^\times$.

Proof. (a) Now that Fact 3 has ruled out the vertices in $V_q^\times \cup V_s^\times$, the only possibility left is to take all the h codewords in $V_p^\times$.

(b) Let C be an optimal r-LD code in G^+, and let $|C \cap (V_s^\times \setminus \{\alpha\})| = X$ and $|C \cap V_p^\times| = Y$. If $Y = |V_p^\times|$, we are done, so we assume that $Y \leq h-1$. How does C r-separate the $2^h-(r-1)h-2-X$ vertices in $V_s^\times \setminus \{\alpha\}$ that need to be r-separated? Depending on its distance to α, a vertex outside $V^\times$ r-dominates either α alone, or all the vertices in $V_s^\times$ plus all the vertices $q_{i,j}$, $1 \leq i \leq h$, for some $j \geq 1$. This means that no outside codeword can r-separate the vertices in $V_s^\times \setminus \{\alpha\}$: this must be an inside-$V^\times$ job. But the vertices in $V_q^\times \cup V_s^\times$ cannot do it either, because every such vertex r-dominates all the vertices in $V_s^\times \setminus \{\alpha\}$; so the Y codewords in $V_p^\times$ must do it, and, according to whether the vertices in $V_s^\times \setminus \{\alpha\}$ are r-dominated by other codewords or not, we must have $2^Y - \varepsilon \geq 2^h - (r-1)h - 2 - X$, with $\varepsilon = 0$ or 1; since $Y \leq h-1$, this implies

$$2^{h-1} \leq (r-1)h + 2 + X - \varepsilon. \tag{4}$$

For $r \geq 2$ and $h = 2r+1$, the study of (4) shows that necessarily $X \geq h+2$. What is the role of these (at least) $h+2$ codewords belonging to $V_s^\times \setminus \{\alpha\}$?

(a) They contribute to r-dominate and r-separate some vertices in $V^+ \setminus V^\times$. From this perspective, all the vertices in $V_s^\times \setminus \{\alpha\}$ have an equivalent role towards $V^+ \setminus V^\times$. So one codeword in $V_s^\times \setminus \{\alpha\}$ is sufficient for this task.

(b) They contribute to r-dominate and r-separate some vertices in $V^\times$, and they themselves need not be r-separated from other vertices by the code; but we have already seen (Fact 1) that if we take the h vertices in $V_p^\times$ as codewords, then we can take care of all the vertices in $V^\times$.

(c) They contribute to r-separate some vertices in $V^+ \setminus V^\times$ from some vertices in $V^\times$. But the h vertices in $V_p^\times$ r-dominate all the vertices inside $V^\times$ and no vertex outside $V^\times$.

Therefore, if we take h codewords in $V_p^\times$ and one codeword in $V_s^\times \setminus \{\alpha\}$, we can do, with respect to the whole graph G^+, at least as well as with $X + Y \geq X \geq h+2$ codewords, contradicting the optimality of C. $\Diamond$

The following lemma and its obvious corollary will be used for Proposition 21. They characterize the vertices belonging to at least one optimal r-LD code, through the comparison of two 1-location-domination numbers.

Lemma 11 *Let $G = (V, E)$ be a graph. For a given vertex $\alpha \in V$, we consider the following graph: $G_\alpha = (V_\alpha, E_\alpha)$, with*

$$V_\alpha = V \cup \{\beta_i : 1 \leq i \leq 6\},$$

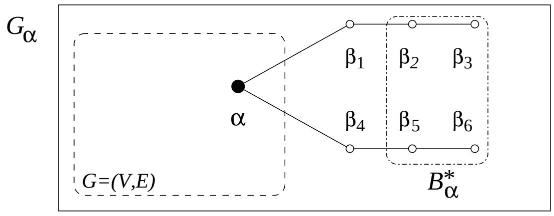

Figure 4: The graphs G and G_α for Lemma 11.

$$E_\alpha = E \cup \{\alpha\beta_i : i \in \{1,4\}\} \cup \{\beta_i\beta_{i+1} : i \in \{1,2,4,5\}\},$$

where for $i \in \{1,\ldots,6\}$, $\beta_i \notin V$ (see Figure 4). Then α belongs to at least one optimal 1-locating-dominating code in G if and only if $LD_1(G) = LD_1(G_\alpha) - 2$.

Proof. Let $B_\alpha = \{\beta_i : 1 \leq i \leq 6\}$, and $B^*_\alpha = \{\beta_i : i \in \{2,3,5,6\}\}$.

First, we prove that α belongs to every optimal 1-LD code C_α in G_α: assume on the contrary that $\alpha \notin C_\alpha$; then obviously $|C_\alpha \cap B_\alpha| \geq 2+2$, and $(C_\alpha \setminus (C_\alpha \cap B_\alpha)) \cup \{\alpha, \beta_2, \beta_5\}$ is a 1-LD code in G_α, with fewer elements than C_α, a contradiction. So $\alpha \in (C_\alpha \cap V)$.

(a) Assume that α belongs to at least one optimal 1-LD code C in G. Then $C_\alpha = C \cup \{\beta_2, \beta_5\}$ is obviously 1-LD in G_α, and $LD_1(G_\alpha) \leq |C_\alpha| = LD_1(G) + 2$.

Consider now an optimal 1-LD code C_α in G_α. Obviously, we have $|C_\alpha \cap B^*_\alpha| \geq 1+1$, and so, if we set $C = C_\alpha \cap V$, we have: $|C| \leq LD_1(G_\alpha) - 2$. We have already established that $\alpha \in C_\alpha$, and thus $\alpha \in C$; now, we can see that it is sufficient, for any optimal 1-LD code in G_α, to have two codewords in B_α, namely β_2 and β_5, and therefore $|C| \geq LD_1(G_\alpha) - 2$. Now, no codeword in $C_\alpha \setminus C$ 1-dominates any vertex in V, and necessarily C is a 1-LD code in G, which proves that $LD_1(G) \leq |C| = LD_1(G_\alpha) - 2$.

So we can conclude that if α belongs to at least one optimal 1-LD code in G, then $LD_1(G) = LD_1(G_\alpha) - 2$.

(b) Assume now that $LD_1(G) = LD_1(G_\alpha) - 2$. Consider an optimal 1-LD code C_α in G_α, and let $C = C_\alpha \cap V$. Then $\alpha \in C_\alpha$ and, exactly as before in (a), $\alpha \in C$, $C_\alpha = C \cup \{\beta_2, \beta_5\}$, C is a 1-LD code in G and its size is $LD_1(G_\alpha) - 2 = LD_1(G)$, i.e., it is optimal (and contains α). $\Diamond$

Corollary 12 *Let $r \geq 1$ be any integer, G be a graph containing a vertex α, and G^r be the r-th power of G. We construct the graph $(G^r)_\alpha$ in the same way as in the previous lemma for G. Then α belongs to at least one optimal r-locating-dominating code in G if and only if $LD_1(G^r) = LD_1((G^r)_\alpha) - 2$.*

Proof. Use Lemmas 5(a) and 11. $\Diamond$

In the following proposition, we shall only use the fact that LDC_1 belongs to *NP*, for Propositions 21 and 22.

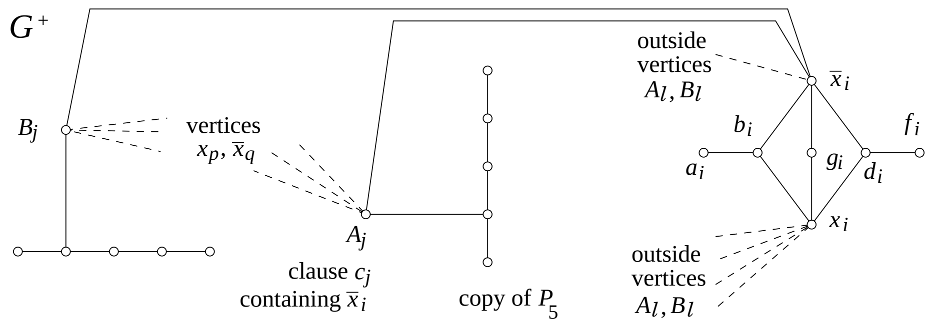

Figure 5: For $r = 1$, the graph G^+, constructed from a set of clauses.

Proposition 13 *[10, for $r = 1$], [13] Let $r \geq 1$ be any integer. The decision problem LDC_r is NP-complete.* $\Diamond$

The proofs of Proposition 13 do not treat however the problem of the uniqueness of a solution.

3.2 Uniqueness of Locating-Dominating Code

3.2.1 From U-SAT to U-LDC$_1$ and U-OLDC$_1$

Theorem 14 *There exists a polynomial reduction from U-SAT to U-LDC$_1$ and to U-OLDC$_1$: U-SAT $\rightarrow_p$ U-LDC$_1$ and U-SAT $\rightarrow_p$ U-OLDC$_1$.*

Proof. We give a polynomial reduction starting from an instance of U-SAT, that is, a collection $\mathcal{C}$ of m clauses over a set $\mathcal{X}$ of n variables.

For each variable $x_i \in \mathcal{X}$, $1 \leq i \leq n$, we take the graph $G_i = (V_i, E_i)$ defined in Lemma 9, identifying the literals $x_i, \overline{x}_i$ to the vertices $x_i, \overline{x}_i$. For each clause c_j, containing ε_j literals, $\varepsilon_j \geq 2$, we create two vertices, A_j and B_j, and we link them to the ε_j vertices corresponding, in the graphs G_i, to the literals of c_j. We also take a copy of P_5, $P_5(A_j) = A_{j,1}A_{j,2}A_{j,3}A_{j,4}A_{j,5}$, and link A_j to $A_{j,2}$. We do the same for B_j and a second copy of P_5, $P_5(B_j) = B_{j,1}B_{j,2}B_{j,3}B_{j,4}B_{j,5}$.

We call this graph G^+, see Figure 5. The order of G^+ is $7n + 12m$. Because the extremities of the copies of P_5 must be 1-dominated by a codeword, and thanks to Lemma 9(d), we have: $LD_1(G^+) \geq 4m + 3n$. We set $k = 4m + 3n$.

We claim that there is a unique solution to SAT if and only if there is a unique optimal 1-LD code in G^+, and if and only if there is a unique 1-LD code of size at most k in G^+.

(1) Assume first that there is a unique truth assignment satisfying all the clauses. We construct the following code C: for $i \in \{1, \ldots, n\}$, among the vertices $x_i \in V_i$, $\overline{x}_i \in V_i$, we put in C the vertex x_i if the literal x_i has been set TRUE, the vertex $\overline{x}_i$ if the literal x_i is FALSE, and we add b_i and d_i, as well as the second and fourth vertices in each of the copies of P_5. Then C is a 1-LD code in G^+: thanks to our preliminary observations (Lemmas 8 and 9), the only thing that remains to be checked is that for all $j \in \{1, \dots, m\}$, the code C 1-separates A_j and $A_{j,1}$, B_j and $B_{j,1}$, and this is so because there is at least one true literal in the clause c_j, which means that A_j and B_j are 1-dominated by at least one codeword of type x_i or $\overline{x}_i$.

Moreover, $|C| = 3n + 4m = k$, which proves that it is optimal, and no vertex A_j nor B_j is a codeword. This implies that, once we have decided between x_i and $\overline{x}_i$, we have *no choice left* inside G_i if we want a code of size k (optimal): we *must* take b_i and d_i, because neither x_i nor $\overline{x}_i$ is 1-dominated by any outside codeword; the same is true for the copies of P_5, which *must* each contain their second and fourth vertices as only codewords.

Why is C unique? Suppose on the contrary that C^* is another 1-LD code of size $k = 3n + 4m$ in G^+. Then $|C^* \cap V_i| = 3$ for all $i \in \{1, \dots, n\}$, and at most one of x_i and $\overline{x}_i$ is in C^*. Also, no vertex A_j or B_j is a codeword, and each copy of P_5 contains exactly two codewords, which are necessarily the second and the fourth ones. As another consequence, at least one of x_i and $\overline{x}_i$ is a codeword, because no codeword 1-separates them, and so exactly one of them belongs to C^*. This defines a valid truth assignment for $\mathcal{X}$, by setting $x_i =$T if $x_i \in C^*$, $x_i =$F if $\overline{x}_i \in C^*$. Since $C \neq C^*$, this assignment is different from the assignment used to build C. But the fact that, for all j, C^* 1-separates A_j and $A_{j,1}$, B_j and $B_{j,1}$, shows that there is a codeword x_i or $\overline{x}_i$ 1-dominating A_j and B_j, which means that the clause c_j is satisfied. Therefore, we have a second assignment satisfying the instance of SAT, a contradiction. We can conclude that both problems, U-LDC$_1$ and U-OLDC$_1$, also receive the answer YES.

(2) Assume next that the answer to U-SAT is NO: this may be either because no truth assignment satisfies the instance, or because at least two assignments do; in the latter case, this would lead, using the same argument as previously, to at least two (optimal) 1-LD codes (of size k), and a NO answer to both U-LDC$_1$ and U-OLDC$_1$. So we are left with the case when the set of clauses $\mathcal{C}$ cannot be satisfied. This implies that no 1-LD code of size k exists, for the same reason as in the previous paragraph with C^*; this suffices to prove that we have also a NO answer to U-LDC$_1$, but we have to go further for U-OLDC$_1$. Assume then that C is an optimal 1-LD code of unknown size $|C| > 4m + 3n$. For $1 \leq j \leq m$, let $\mathcal{A}_j = C \cap \{A_j, A_{j,i} : 1 \leq i \leq 5\}$ and $\mathcal{B}_j = C \cap \{B_j, B_{j,i} : 1 \leq i \leq 5\}$.

Suppose first that there is a j_0 such that A_{j_0} and B_{j_0} are not 1-dominated by any codeword x_i nor any codeword $\overline{x}_i$. Then $|\mathcal{A}_{j_0}| > 2$, and actually this set has size exactly three; the same is true for $\mathcal{B}_{j_0}$. Now we have several optimal codes, because $C \cap (\mathcal{A}_{j_0} \cup \mathcal{B}_{j_0})$ can be equal to

$\{A_{j_0}, A_{j_0,2}, A_{j_0,4}, B_{j_0}, B_{j_0,2}, B_{j_0,4}\}$, or

$\{A_{j_0,1}, A_{j_0,2}, A_{j_0,4}, B_{j_0}, B_{j_0,2}, B_{j_0,4}\}$, or

$\{A_{j_0}, A_{j_0,2}, A_{j_0,4}, B_{j_0,1}, B_{j_0,2}, B_{j_0,4}\}$,
—but in general, not $\{A_{j_0,1}, A_{j_0,2}, A_{j_0,4}, B_{j_0,1}, B_{j_0,2}, B_{j_0,4}\}$, for this might affect the vertices x_i, $\overline{x}_i$ to which A_{j_0} and B_{j_0} are linked.

So from now on, we assume that all vertices A_j, B_j are 1-dominated by at least one codeword x_i or $\overline{x}_i$. Because the set of clauses cannot be satisfied, it is impossible that for all i, exactly one of x_i and $\overline{x}_i$ is a codeword, for this would lead to a valid truth assignment which would satisfy all the clauses. So there is a subscript i_0 such that either both x_{i_0} and $\overline{x}_{i_0}$ are codewords, or none is a codeword. If both are codewords, then $C \cap V_{i_0}$ contains x_{i_0}, $\overline{x}_{i_0}$ and can contain any combination with exactly one codeword among a_{i_0}, b_{i_0} and exactly one among d_{i_0} and f_{i_0}, yielding at least four optimal solutions. If none of $x_{i_0}, \overline{x}_{i_0}$ is a codeword, then they are 1-separated by some codeword(s) A_j, B_j; moreover, g_{i_0}, which must belong to C, 1-dominates both x_{i_0}, $\overline{x}_{i_0}$. Then again, we can have any of the four combinations with one codeword among a_{i_0}, b_{i_0} and one among d_{i_0} and f_{i_0}.

So in all cases, we do not have a unique optimal 1-LD code. $\diamondsuit$

3.2.2 Extension to $r \geq 2$

It seems difficult to go directly from $r = 1$ to the general case $r \geq 2$, and we start again from U-SAT, which does not change the final result; see [13, Rem. 5] about this possible difficulty.

Theorem 15 *Let $r \geq 2$ be any integer. There exists a polynomial reduction from U-SAT to U-LDC$_r$ and to U-OLDC$_r$: U-SAT $\to_p$ U-LDC$_r$ and U-SAT $\to_p$ U-OLDC$_r$.*

Proof. We give a polynomial reduction starting from an instance of U-SAT, i.e., a collection $\mathcal{C}$ of m clauses over a set $\mathcal{X}$ of n variables.

We take the graph G^+ constructed in the proof of Theorem 14 (cf. Figure 5), and rename it $G_I = (V_I, E_I)$, for Intermediate graph. Then, for each edge $e = uv \in E_I$, we "paste" $r-1$ copies of the graph $G^\times$ constructed for Proposition 10 (cf. Figure 3), by deleting the edge $e = uv$ and creating the edges $u\alpha_1$, $\alpha_1\alpha_2$, …, $\alpha_{r-1}v$, where the α_i's are copies of the vertex α in $G^\times$, see Figure 6: we shall say that the edge e is *dilated*. We denote by G^+ the graph thus constructed. Since r, hence $h = 2r+1$, is fixed, the fact that $G^\times$ has order $2^h + h - 1$ does not affect the polynomiality of our construction with respect to $n + m$. We set $k = 3n + 4m + (r-1)h|E_I|$.

The use of copies of $G^\times$ can be seen as a way of putting at distance r, in the graph G^+, the vertices which are at distance one in G_I, so that the vertices in V_I will behave with respect to each other in a way very similar to the case $r = 1$. It is still true that, in addition to at least h codewords taken in each copy of $G^\times$, at least three codewords are necessary in order to deal with the vertices in each V_i, and that at least two are necessary to cope with every copy of $\{x_1, \ldots, x_5\}$, the set of vertices in P_5. Consequently, any

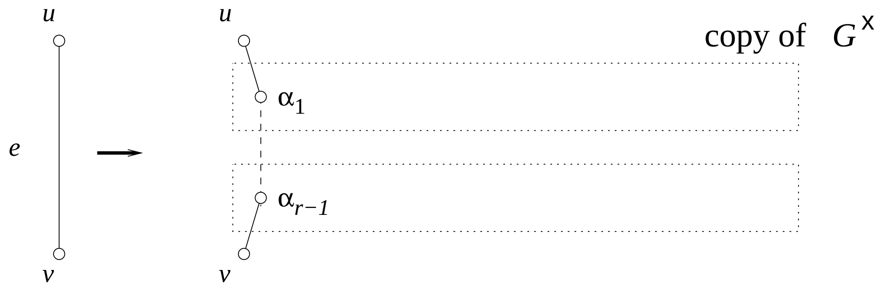

Figure 6: How the edge $e = uv \in E_I$ is dilated in the proof of Theorem 15.

optimal r-LD code in G^+ has size at least $k = 3n + 4m + (r-1)h|E_I|$, the three terms corresponding respectively to (a) the sets V_i, $1 \leq i \leq n$, (b) the $2m$ copies of P_5 (which are now dilated copies), and (c) the $(r-1)$ copies of the graph $G^\times$ on each edge of the intermediate graph G_I.

One role of the codewords is to deal with the vertices in V_I, that is, if these are not codewords themselves, to r-dominate them, and to r-separate between them —the domination and separation inside the copies of $G^\times$ and the separation between vertices in V_I and vertices in the copies of $G^\times$ are already performed by the copies of the cycle $\mathcal{C}_h$, which are necessarily included in any optimal r-LD code in G^+, as already observed.

We can also make the following useful remark.

Remark 16 *Let C be any optimal r-locating-dominating code in G^+, $e = uv$ be any edge in E_I,and $G^\times$ be one of the copies pasted on e. If z is a codeword belonging to $G^\times$ and not to its cycle $\mathcal{C}_h$, then $(C \setminus \{z\}) \cup \{u\}$ or $(C \setminus \{z\}) \cup \{v\}$ is also an optimal r-locating-dominating code in G^+.*

Indeed, if (a) z r-dominates neither u nor v, then it can be spared and C is not optimal; if (b) z r-dominates u, not v, i.e., it r-separates u and v (and z cannot r-dominate any other vertex in V_I), then, when u or v becomes a codeword, u and v need not be r-separated anymore; if (c) z r-dominates both u and v, these two vertices will remain r-dominated by a common codeword, u or v. In all cases, the fact that other vertices in V_I can now be r-dominated by the substitute codeword (u or v) does not change anything (cf. Lemma 6(a)).

We claim that there is a unique solution to SAT if and only if there is a unique optimal r-LD code in G^+, and if and only if there is a unique r-LD code of size at most k in G^+.

(1) Assume first that there is a unique truth assignment satisfying all the clauses. We construct the following code C: for $i \in \{1, \ldots, n\}$, among the vertices $x_i \in V_i$, $\overline{x}_i \in V_i$, we put in C the vertex x_i if the literal x_i has been set TRUE, the vertex $\overline{x}_i$ if the literal x_i is FALSE, and we add b_i and d_i, as well as the second and fourth vertices in each of the (dilated) copies of P_5. We add the cycle $\mathcal{C}_h$ in each copy of $G^\times$. Then C is an r-LD code in G^+: thanks to our preliminary observations, the only thing that remains to be checked is that for all $j \in \{1, \ldots, m\}$, the code C r-separates A_j and $A_{j,1}$, B_j and $B_{j,1}$, and this is so because there is at least one true literal in the clause c_j, which means that A_j and B_j are r-dominated by at least one codeword of type x_i or $\overline{x}_i$: everything develops exactly as in the case $r = 1$.

Moreover, $|C| = k$, which proves that it is optimal, and no vertex A_j nor B_j is a codeword. This implies that, once we have decided between x_i and $\overline{x}_i$, we have no choice left inside G_i: we *must* take b_i and d_i, because neither x_i nor $\overline{x}_i$ is r-dominated by any outside codeword. We have no choice in the copies of P_5 either: no pair of vertices in copies of $G^\times$ can r-dominate the first and last vertices, and r-separate the first, third and last, at the same time: only the second and fourth vertices can perform this.

Why is C unique? Suppose on the contrary that C^* is another (optimal) r-LD code (of size k) in G^+. Then each copy of $G^\times$ intersects C^* on exactly h vertices which are the vertices of the cycle $\mathcal{C}_h$; also, $|C^* \cap V_i| = 3$ for all $i \in \{1, \ldots, n\}$, and at most one of x_i and $\overline{x}_i$ is in C^*. Moreover, no vertex A_j nor B_j is a codeword. As a consequence, at least one of x_i and $\overline{x}_i$ is a codeword, because no codeword r-separates them, and so exactly one of them belongs to C^*. This defines a valid truth assignment for $\mathcal{X}$, by setting $x_i =$T if $x_i \in C^*$, $x_i =$F if $\overline{x}_i \in C^*$. Since $C \neq C^*$, this assignment is different from the assignment used to build C. But the fact that, for all j, C^* r-separates A_j and $A_{j,1}$, B_j and $B_{j,1}$, shows that there is a codeword x_i or $\overline{x}_i$ r-dominating A_j and B_j, which means that the clause is satisfied. Therefore, we have a second assignment satisfying the instance of SAT, a contradiction.

(2) Assume next that the answer to U-SAT is NO: this may be either because no truth assignment satisfies the instance, or because at least two assignments do; in the latter case, this would lead, using the same argument as previously, to at least two optimal r-LD codes (of size k), and a NO answer to U-LDC$_r$ and U-OLDC$_r$. So we are left with the case when the set of clauses $\mathcal{C}$ cannot be satisfied. This implies that no r-LD code of size k exists; this ends the case of U-LDC$_r$ but we have to go on with U-OLDC$_r$: assume that C is an optimal r-LD code of unknown size $|C| > k$, with at least three codewords to deal with each V_i, at least two codewords to deal with each copy of P_5, at least h codewords in each copy of $G^\times$, and possibly vertices of type A_j, B_j. By Remark 16, if there is a codeword belonging to a copy of $G^\times$ and not to its cycle $\mathcal{C}_h$, then we have at least two optimal r-LD codes, and we are done. So from now on we assume that no copy of $G^\times$ contains codewords outside the cycle $\mathcal{C}_h$. Then C contains at least three codewords in each V_i, at least two codewords in each copy of P_5, exactly h codewords in each copy of $G^\times$, and possibly vertices of type A_j and B_j. Now the argument of the case $r = 1$ (Theorem 15) can be repeated almost word for word: first, we can exclude that there is a vertex A_{j_0} not r-dominated by any codeword x_i or $\overline{x}_i$; then we deal with the cases when both x_i and $\overline{x}_i$ are codewords, and when none of them is. In all cases, we have more than one optimal r-LD code in G^+. $\diamondsuit$

Corollary 17 *Let $r \geq 1$ be any integer. The decision problems U-LDC$_r$ and U-OLDC$_r$ are NP-hard.*

Proof. Because U-SAT is *NP*-hard (Proposition 2). $\diamondsuit$

Proposition 18 *Let $r \geq 2$ and $q \geq 1$ be any integers. There is a polynomial reduction from U-LDC$_{qr}$ to U-LDC$_q$ and from U-OLDC$_{qr}$ to U-OLDC$_q$: U-LDC$_{qr}$ $\to_p$ U-LDC$_q$ and U-OLDC$_{qr}$ $\to_p$ U-OLDC$_q$.*

As a particular case, we have U-LDC$_r$ $\to_p$ U-LDC$_1$ and U-OLDC$_r$ $\to_p$ U-OLDC$_1$.

Proof. Let (G, k) be an instance of U-LDC$_{qr}$ and G be an instance of U-OLDC$_{qr}$, for $r \geq 2$ and $q \geq 1$. The instance for U-LDC$_q$ is simply (G^r, k), and G^r for U-OLDC$_q$, where G^r is the r-th power of G: obviously, by Lemma 5(a), there is a unique q-LD code of size k in G^r if and only if there is a unique qr-LD code of size k in G, and there is a unique optimal q-LD code in G^r if and only if there is a unique optimal qr-LD code in G. $\diamondsuit$

3.2.3 An Upper Bound for the Complexity of U-LDC$_r$

Theorem 19 *There exists a polynomial reduction from U-LDC$_1$ to U-SAT: U-LDC$_1$ $\to_p$ U-SAT.*

Proof. In [17, Rem. 10] we developed a general argument for this kind of reduction, with three types of clauses, one type for the description of the specific problem, here the fact that we want the code to be 1-LD, one type for the fact that we want a code of size at most k, and one type to break the multiple solutions. The same method will be applied for Theorem 34, with 1-identifying codes.

We start from an instance of U-LDC$_1$: a graph $G = (V, E)$ and an integer k, with $V = \{x^1, \ldots, x^{|V|}\}$; we assume that $|V| \geq 3$. We create the set of $k|V|$ variables $\mathcal{X} = \{x^i_m : 1 \leq i \leq |V|, 1 \leq m \leq k\}$ and the following clauses:

(a1) for each vertex $x^i \in V$ with neighbours $x^{n_1}, \ldots, x^{n_s}$ (where $s = s(x^i)$ is the degree of x^i), we take the clause of size $k(s+1)$:

$$c_{x^i} = \{x^i_1, x^i_2, \ldots, x^i_k, x^{n_1}_1, x^{n_1}_2, \ldots, x^{n_1}_k, x^{n_2}_1, \ldots, x^{n_2}_k, \ldots, x^{n_s}_1, \ldots, x^{n_s}_k\};$$

(a2) for each pair of vertices $x^i \in V$, $x^j \in V$, we consider the set $B_1(x^i)\Delta B_1(x^j) = \{x^{h_1}, x^{h_2}, \ldots, x^{h_\ell}\}$ (where ℓ depends on x^i and x^j) and we construct the clause $c_{x^i x^j}$:

$$\{x^i_1, x^i_2, \ldots, x^i_k, x^j_1, \ldots, x^j_k, x^{h_1}_1, x^{h_1}_2, \ldots, x^{h_1}_k, x^{h_2}_1, \ldots, x^{h_2}_k, \ldots, x^{h_\ell}_1, \ldots, x^{h_\ell}_k\};$$

we shall say that $\{x_1^i, x_2^i, \dots, x_k^i, x_1^j, \dots, x_k^j\}$ is the first part of $c_{x^ix^j}$ and $\{x_1^{h_1}, x_2^{h_1}, \dots, x_k^{h_1}, x_1^{h_2}, \dots, x_k^{h_2}, \dots, x_1^{h_\ell}, \dots, x_k^{h_\ell}\}$ its second part, which exists only when $\ell > 0$ and may contain variables also appearing in the first part, which is unimportant;

(b1) for $1 \le m \le k$ and $1 \le h < \ell \le |V|$, we construct clauses of size two: $\{\overline{x}_m^h, \overline{x}_m^\ell\}$;

(b2) for $1 \le m < s \le k$ and $1 \le h \le |V|$, we construct clauses of size two: $\{\overline{x}_m^h, \overline{x}_s^h\}$;

(c) for $1 \le m < k$ and $1 < \ell \le |V|$, for $1 \le h < \ell$ and $m < s \le k$, we construct clauses of size two: $\{\overline{x}_m^\ell, \overline{x}_s^h\}$.

All these clauses constitute the instance of U-SAT. Note that the number of variables and clauses is polynomial with respect to the order of G, since we may assume that $k \le |V|$.

Assume that we have a unique 1-LD code of size k in G, $C = \{x^{p_1}, x^{p_2}, \dots, x^{p_k}\}$, with $p_1 < p_2 < \dots < p_k$. We can see that C is optimal (otherwise, any optimal 1-LD code contradicts our uniqueness assumption). Define the assignment $\mathcal{A}_1$ by $\mathcal{A}_1(x_q^{p_q}) = \mathrm{T}$ for $1 \le q \le k$, and all the other variables are set FALSE by $\mathcal{A}_1$. We claim that this assignment satisfies all the clauses; indeed:

(a1) at least one among x^i and its neighbours is a codeword, so the clause c_{x^i} is satisfied by $\mathcal{A}_1$.

(a2) (i) If at least one of x^i or x^j belongs to C, say $x^i = x^{p_q} \in C$, then the variable $x_q^{p_q} = x_q^i$ has been set True by $\mathcal{A}_1$ and the first part of the clause $c_{x^ix^j}$, hence the whole clause, is satisfied. (ii) If neither x^i nor x^j belongs to C, then, using the characterization given by (1), we can see that at least one x^{h_m} belongs to C, which guarantees that the second part of $c_{x^ix^j}$ is satisfied.

(b1) If a clause $\{\overline{x}_m^h, \overline{x}_m^\ell\}$ is not satisfied for some m, h, ℓ, this means that $\mathcal{A}_1(x_m^h) = \mathcal{A}_1(x_m^\ell) = \mathrm{T}$, i.e., two different vertices are the m-th element in C.

(b2) If $\{\overline{x}_m^h, \overline{x}_s^h\}$ is not satisfied, then x^h appears at least twice in C.

(c) If $\{\overline{x}_m^\ell, \overline{x}_s^h\}$ is not satisfied for some m, ℓ, with $h < \ell$ and $m < s$, then $\mathcal{A}_1(x_m^\ell) = \mathcal{A}_1(x_s^h) = \mathrm{T}$. This means that $x^\ell = x^{p_m}$ and $x^h = x^{p_s}$; so $\ell = p_m$, $h = p_s$. Now $h < \ell$ implies that $p_s < p_m$, but $m < s$ implies that $p_m < p_s$, a contradiction.

Is $\mathcal{A}_1$ unique? Assume on the contrary that another assignment, $\mathcal{A}_2$, also satisfies the constructed instance of U-SAT. We construct a new code C^+ by putting in C^+ the vertex x^h as soon as some variable x_m^h is set TRUE by $\mathcal{A}_2$.

Now the satisfaction, by $\mathcal{A}_2$, of the clause c_{x^i} in (a1) proves that every vertex in V is 1-dominated by C^+; the satisfaction of $c_{x^ix^j}$ from (a2) means that at least one vertex among $x^i, x^j, x^{h_1}, \dots, x^{h_\ell}$ belongs to C^+. So for every pair of vertices x^i, x^j, either one of them is in the code, or an element in

$B_1(x^i)\Delta B_1(x^j)$ is in the code. So every pair of non-codewords is 1-separated by C^+.

Therefore, we have just proved that C^+ is a 1-LD-code.

Using (b1) for $\mathcal{A}_2$, we can see that for each $m \in \{1, \ldots, k\}$ there is at most one variable with subscript m that is set TRUE by $\mathcal{A}_2$; this means that we have constructed a 1-LD code with (at most) k elements. Since such a code is unique by assumption, we can see that $\mathcal{A}_1$ and $\mathcal{A}_2$ "selected" the same k codewords: for each $p_q \in \{p_1, \ldots, p_k\}$, we already know that there is exactly one variable, $x_q^{p_q}$, set TRUE by $\mathcal{A}_1$, and, using (b2) after (b1) for $\mathcal{A}_2$, exactly one variable, say $x_t^{p_q}$, set TRUE by $\mathcal{A}_2$. It is now time to use (c) in order to prove that $q = t$ for every p_q, so that $\mathcal{A}_1$ and $\mathcal{A}_2$ actually coincide: indeed, assume on the contrary that for some $q \in \{1, \ldots, k\}$, we have $q \neq t$; we treat the case $1 \leq q < t \leq k$, the case $1 \leq t < q \leq k$ being analogous. If we consider the subscripts smaller than t, there must be one, say v, such that there is a superscript $p_u > p_q$ verifying $\mathcal{A}_2(x_v^{p_u}) = $T. Now the clause $\{\overline{x}_v^{p_u}, \overline{x}_t^{p_q}\}$ from (c) is not satisfied by $\mathcal{A}_2$, a contradiction.

So a YES answer to U-LDC$_1$ leads to a YES answer to U-SAT. Assume now that the answer to U-LDC$_1$ is negative. If it is negative because there are at least two 1-LD codes of size k, then we have at least two assignments satisfying the instance of U-SAT: we have seen above how to construct a suitable assignment from a 1-LD code, and different 1-LD codes obviously lead to different assignments. On the other hand, if there is no 1-LD code of size k, then there can be no assignment satisfying U-SAT, because such an assignment would give a 1-LD code of size k, as we have seen above when dealing with $\mathcal{A}_2$. So in both cases, a NO answer to U-LDC$_1$ implies a NO answer to U-SAT. $\diamondsuit$

By Proposition 18 or its corollary, this immediately implies that there is a polynomial reduction from U-LDC$_r$ to U-SAT.

Theorem 20 *Let $r \geq 1$ be any integer. The problem U-LDC$_r$ has complexity equivalent to that of U-SAT.*

As a consequence, U-LDC$_r$ belongs to the class DP. $\diamondsuit$

Note that it could have been shown directly that U-LDC$_r$ belongs to *DP*.

3.2.4 Two Upper Bounds for the Complexity of U-OLDC$_r$

In [17], we give, for two problems structurally similar to U-OLDC$_r$, two upper bounds, the first one being weaker but constructive. We do not give the proofs of the following two results but refer to [17] instead. These proofs use Lemma 11, Corollary 12 and Proposition 13.

Proposition 21 *For $r \geq 1$, the decision problem U-OLDC$_r$ belongs to the class P^{NP}. In case of a YES answer, one can give the only optimal r-locating-dominating code within the same complexity.* $\diamondsuit$

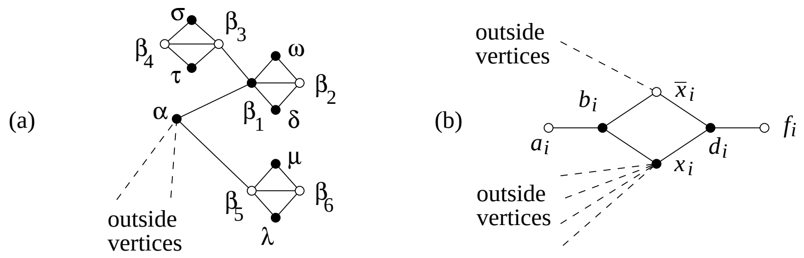

Figure 7: (a) The graph $G^{\times}$ of Lemma 23. Black vertices belong to any 1-identifying code in $G^{\times}$. (b) The graph G_i of Lemma 24, with an optimal 1-identifying code (black vertices).

Proposition 22 *For $r \geq 1$, the decision problem U-OLDC$_r$ belongs to L^{NP}.* $\diamondsuit$

4 Identifying Codes

The structure of this Section and its results are the same as Section 3 for LD-codes, although the preliminary graphs and arguments are quite different technically.

4.1 Preliminary Results

Lemma 23 will be used in the proof of Theorem 30, and Lemma 24 in the proofs of Theorems 30 and 31.

Lemma 23 *Let $G^{\times} = (V^{\times}, E^{\times})$ be the following graph:*

$$V^{\times} = \{\alpha, \beta_1, \beta_2, \beta_3, \beta_4, \beta_5, \beta_6, \omega, \delta, \sigma, \tau, \lambda, \mu\},$$

$$E^{\times} = \{\alpha\beta_1, \beta_1\beta_2, \beta_1\delta, \beta_1\omega, \beta_2\delta, \beta_2\omega, \beta_1\beta_3, \beta_3\beta_4, \beta_3\sigma, \beta_3\tau, \beta_4\sigma, \beta_4\tau\} \cup$$
$$\cup \{\alpha\beta_5, \beta_5\beta_6, \beta_5\lambda, \beta_5\mu, \beta_6\lambda, \beta_6\mu\},$$

see Figure 7(a). Then $i_1(G^{\times}) = 8$, any 1-identifying code in $G^{\times}$ contains the set of vertices $C = \{\alpha, \beta_1, \omega, \delta, \sigma, \tau, \lambda, \mu\}$, and C is the only optimal 1-identifying code in $G^{\times}$.

If $G^{\times}$ is plunged in a larger graph G^{+}, with only α linked to the outside, then every 1-identifying code in G^{+} contains C; the outside neighbours of α are 1-covered, not 1-separated, by α, and they are 1-separated from $V^{\times}$ by C.

Proof. Straightforward: α is the only vertex 1-separating β_5 and β_6; the same is true about β_1, for β_3, β_4; about ω, for β_2, δ; about δ, for β_2, ω; about σ, for β_4, τ; about τ, for β_4, σ; about λ, for β_6, μ; and about μ, for β_6, λ. So these eight vertices belong to any 1-identifying code, and since they obvioulsy constitute a 1-identifying code, this is the only optimal 1-identifying code.

When we consider G^+, all the above arguments still work. $\diamondsuit$

The statements of the following lemma have been given, although in a different way, in [14, proof of Th. 5.1]; for completeness, we give here the (easy) proof.

Lemma 24 *Let $G_i = (V_i, E_i)$ be the following graph: $V_i = \{x_i, \overline{x}_i, a_i, b_i, d_i, f_i\}$ and $E_i = \{a_i b_i, b_i x_i, b_i \overline{x}_i, x_i d_i, \overline{x}_i d_i, d_i f_i\}$, and C_i be a 1-identifying code in G_i, see Figure 7(b). Then*

(a) At least one of x_i and $\overline{x}_i$ belong to C_i.

(b) At least two more codewords are necessary in C_i, so that $i_1(G_i) \geq 3$.

(c) We have $i_1(G_i) = 3$, and $\{x_i, b_i, d_i\}$ and $\{\overline{x}_i, b_i, d_i\}$ are the only optimal 1-identifying codes in G_i.

(d) If G_i is plunged in a larger graph G^+, with only x_i and $\overline{x}_i$ linked to the outside, then every 1-identifying code in G^+ contains at least three codewords in V_i, and one of them is x_i or $\overline{x}_i$.

Proof. (a) Because a_i and b_i, or d_i and f_i, must be 1-separated by C_i. (b) Because a_i and f_i must be 1-covered by C_i. Alternatively, use (3) in Lemma 4. (c) Assume that it is x_i that belongs to C_i. Then taking a_i and f_i would not 1-cover $\overline{x}_i$, and taking a_i and d_i, or b_i and f_i, would not 1-separate x_i and d_i, or x_i and b_i, respectively; on the other hand, $\{x_i, b_i, d_i\}$ is 1-identifying. (d) Only x_i and $\overline{x}_i$ can be 1-covered by the outside, and a_i and f_i still have to be 1-covered, a_i and b_i, and d_i and f_i still must be pairwise 1-separated by a codeword, requiring at least three inside codewords, one of them being x_i or $\overline{x}_i$. $\diamondsuit$

The *diapason* and *shortened diapason*, introduced in the following lemma and its corollary, will be used in the proof of Theorem 31.

Lemma 25 *[13, Lemma 2.1, Cor. 2.1] Let $r \geq 2$ be any integer. Let $T = \{t_1, t_2, \ldots, t_r\}$, $Y = \{y_1, y_2, \ldots, y_{2r+1}\}$, and $Z = \{z_1, z_2, \ldots, z_{2r+1}\}$. Let Δ be the graph in Figure 8, with vertex set $T \cup Y \cup Z$ and edge set*

$$\{t_i t_{i+1} : i = 1, 2, \ldots, r-1\} \cup \{t_r y_1, t_r z_1\} \cup \{y_i y_{i+1}, z_i z_{i+1} : i = 1, 2, \ldots, 2r\}.$$

(a) The smallest r-identifying code in Δ, C_0, has size $2r+2$ and is unique: it consists of the vertices $y_1, y_2, \ldots, y_r, y_{2r+1}, z_1, z_2, \ldots, z_r$, and z_{2r+1}.

(b) Any r-identifying code in Δ contains at least $2r+2$ elements in $Y \cup Z$; among them, the $2r$ vertices $y_1, y_2, \ldots, y_r$ and $z_1, z_2, \ldots, z_r$ must belong to any r-identifying code.

(c) Consider $r-1$ copies, $\Delta_1, \Delta_2, \ldots, \Delta_{r-1}$, of the graph Δ, and in each copy rename the "first" vertex t_1 by $t_{1,1}, t_{2,1}, \ldots, t_{r-1,1}$, and the other vertices accordingly. Build the graph Ω (cf. Figure 9) by taking these

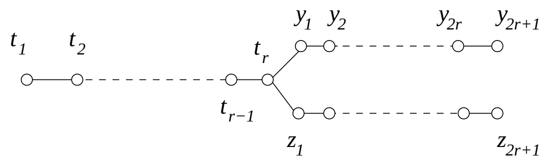

Figure 8: The diapason-graph Δ.

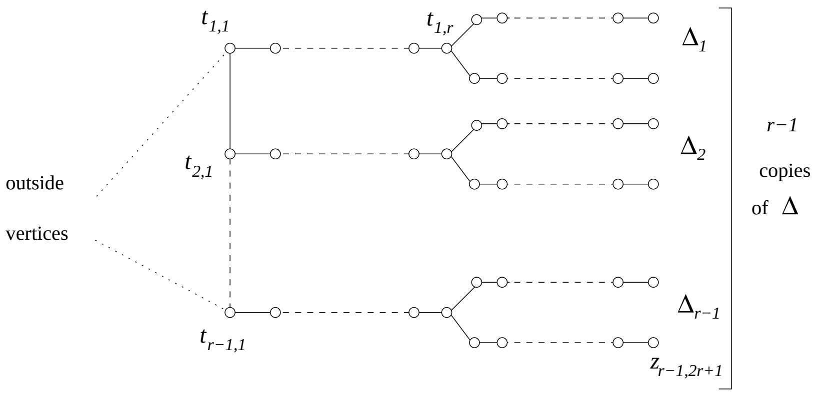

Figure 9: The graph Ω.

$r-1$ copies and adding the edges $t_{1,1}t_{2,1}$, $t_{2,1}t_{3,1}$, $\ldots$, $t_{r-2,1}t_{r-1,1}$. Then the smallest r-identifying code in Ω, C_1, has size $(r-1)(2r+2)$, is unique and consists of $r-1$ copies of the code C_0, one copy of C_0 in each copy of Δ.

(d) If Ω is plunged in a larger graph G^+, with only $t_{1,1}$ and $t_{r-1,1}$ linked to the outside, then every r-identifying code in G^+ contains C_1; no outside vertex is r-covered by C_1. $\diamondsuit$

We call the graph Δ a *diapason*. The sets Y and Z are the *branches*, the set T the *stem*, the vertex t_1 the *foot* of the diapason.

Corollary 26 *If we modify the previous lemma by considering a set T^- with one vertex less: $T^- = \{t_1, t_2, \ldots, t_{r-1}\}$ and t_{r-1} linked to y_1 and z_1, and if we denote by Δ^- the graph thus obtained, then the statements (a) and (b) of Lemma 25 remain true when we replace Δ by Δ^-.*

If Δ^- is plunged in a larger graph G^+, with only t_1 linked to the outside, then every r-identifying code in G^+ contains C_0; the outside neighbours of t_1 are the only outside vertices r-covered by C_0, they are not r-separated from one another by C_0, and they are r-separated by C_0 from all the vertices in Δ^-. $\diamondsuit$

We call the graph Δ^- a *shortened diapason.*

Lemma 27 below, and its corollary, are similar to Lemma 11 and Corollary 12 on r-LD codes: they characterize the vertices belonging to at least one optimal r-identifying code, through the comparison of two 1-identification numbers. They will be used for Proposition 36. They are simplified versions of [15, Lemma 3] and [15, Cor. 4], respectively.

Lemma 27 *Let $G = (V, E)$ be a 1-twin-free graph. For a given vertex $\alpha \in V$, we construct the following graph $G_\alpha = (V_\alpha, E_\alpha)$:*

$$V_\alpha = V \cup \{\beta_1, \beta_2, \delta, \lambda\},\ E_\alpha = E \cup \{\alpha\beta_1, \beta_1\beta_2, \beta_1\delta, \beta_1\lambda, \beta_2\delta, \beta_2\lambda\},$$

where none of the vertices $\beta_1, \beta_2, \delta, \lambda$ belongs to V. Then α belongs to at least one optimal 1-identifying code in G if and only if $i_1(G) = i_1(G_\alpha) - 2$. $\diamondsuit$

Corollary 28 *Let $r \geq 1$ be any integer, G be an r-twin-free graph containing a vertex α, and G^r be the r-th power of G. We construct the graph $(G^r)_\alpha$ in the same way as in the previous lemma for G. Then α belongs to at least one optimal r-identifying code in G if and only if $i_1(G^r) = i_1((G^r)_\alpha) - 2$.* $\diamondsuit$

In the following proposition, we shall only use the fact that IdC_1 belongs to *NP* (see Propositions 36 and 37).

Proposition 29 *[14, for $r = 1$], [13] Let $r \geq 1$ be any integer. The decision problem IdC_r is NP-complete.* $\diamondsuit$

But the proofs for Proposition 29 do not deal with the problem of the uniqueness of a solution.

4.2 Uniqueness of Identifying Code

4.2.1 From U-SAT to U-IdC$_1$ and U-OIdC$_1$

Theorem 30 *There exists a polynomial reduction from U-SAT to U-IdC$_1$ and to U-OIdC$_1$: U-SAT $\to_p$ U-IdC$_1$ and U-SAT $\to_p$ U-OIdC$_1$.*

Proof. This proof is inspired by that of the *NP*-completeness of IdC_1 in [14].

We give a polynomial reduction starting from an instance of U-SAT, that is, a collection $\mathcal{C}$ of m clauses over a set $\mathcal{X}$ of n variables.

For each variable $x_i \in \mathcal{X}$, $1 \leq i \leq n$, we take the graph $G_i = (V_i, E_i)$ defined in Lemma 24. For each clause c_j, containing ε_j literals, $\varepsilon_j \geq 2$, we create two vertices, A_j and B_j, and we link A_j to the ε_j vertices corresponding, in the graphs G_i, to the literals of c_j. Finally we link the vertices A_j and B_j to one copy of the graph $G^\times$ defined in Lemma 23, one different copy for each couple (A_j, B_j), by creating the edges $A_j\alpha$ and $B_j\alpha$ (or rather: we use the j-th copy of α); we call this graph G^+, see Figure 10. The order of G^+ is $6n + 15m$. Note that each pair of vertices A_j, B_j, $1 \leq j \leq m$, is

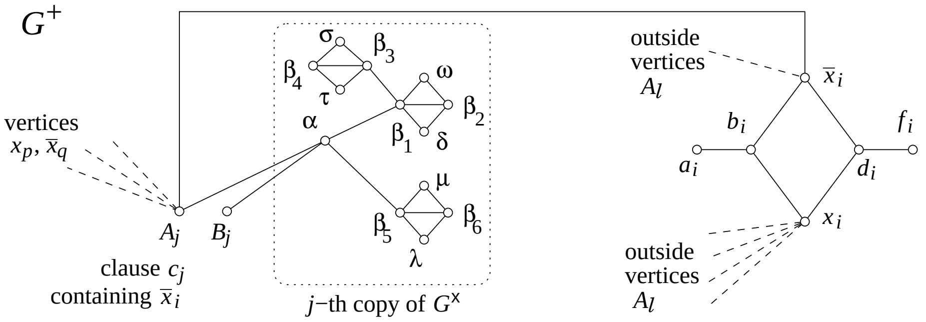

Figure 10: For $r = 1$, the graph G^+, constructed from a set of clauses.

1-covered by a copy of α (which belongs necessarily to any 1-identifying code in G^+, see Lemma 23). By Lemmas 23 and 24, we have $i_1(G^+) \geq 8m + 3n$. We set $k = 8m + 3n$.

We claim that there is a unique solution to SAT if and only if there is a unique optimal 1-identifiying code in G^+, and if and only if there is a unique 1-identifying code of size at most k in G^+.

(1) Assume first that there is a unique truth assignment satisfying all the clauses. We construct the following code C: for $i \in \{1, \ldots, n\}$, among the vertices $x_i \in V_i$, $\overline{x}_i \in V_i$, we put in C the vertex x_i if the literal x_i has been set TRUE, the vertex $\overline{x}_i$ if the literal x_i is FALSE, and we add b_i and d_i. We add all the copies of the vertices α, β_1, ω, δ, σ, τ, λ and μ. Then C is a 1-identifying code in G^+: thanks to all our preliminary observations (Lemmas 23 and 24), the only thing that remains to be checked is that for all $j \in \{1, \ldots, m\}$, the vertices A_j and B_j are 1-separated by C. And this is so because there is at least one true literal in the clause c_j. Moreover, $|C| = k$, which proves that it is optimal. We can also see that, once we have decided between x_i and $\overline{x}_i$, we have no choice left inside G_i: we *must* take b_i and d_i, because neither x_i nor $\overline{x}_i$ is 1-covered by outside codewords, due to the fact that no vertex A_j can be a codeword, by an argument of cardinality.

Why is C unique? Suppose on the contrary that C^* is another 1-identifying code of size k in G^+. Then $|C^* \cap V_i| = 3$ for all $i \in \{1, \ldots, n\}$, and exactly one of x_i and $\overline{x}_i$ is in C^*. This defines a valid truth assignment for $\mathcal{X}$, by setting $x_i =$T if $x_i \in C^*$, $x_i =$F if $\overline{x}_i \in C^*$. Since $C \neq C^*$, this assignment is different from the assignment used to build C. But the fact that C^* 1-separates A_j and B_j for all j shows that there is a codeword x_i or $\overline{x}_i$ 1-covering A_j, which means that the clause is satisfied. Therefore, we have a second assignment satisfying the instance of SAT, a contradiction.

(2) Assume next that the answer to U-SAT is NO: this may be either because no truth assignment satisfies the instance, or because at least two assignments do; in the latter case, this would lead, using the same argument as previously, to at least two optimal 1-identifying codes (of size k), and a NO answer to U-IdC$_1$ and U-OIdC$_1$. So we are left with the case when the set of clauses $\mathcal{C}$ cannot be satisfied. This implies that no 1-identifying code of size k exists; the case U-IdC$_1$ is closed, and, to go on with the problem U-OIdC$_1$, we assume that C is an optimal 1-identifying code of unknown size $|C| > 8m + 3n$. We know that each copy of $G^\times$ contains at least eight codewords, and each G_i at least three codewords. Where can the extra codeword(s) be? Any additional codeword in a copy of $G^\times$ is useless with respect to 1-identification and can be saved. If there are five or six codewords in a G_i, at least one can be saved; assume next that there are four of them: (a) if both x_i and $\overline{x}_i$ are codewords, then, e.g., $C \cap V_i = \{x_i, \overline{x}_i, b_i, d_i\}$ or $C \cap V_i = \{x_i, \overline{x}_i, b_i, f_i\}$ can be part of an optimal solution; (b) if only one of x_i and $\overline{x}_i$, say x_i, is a codeword, then there are also several possibilities for $C \cap V_i$, such as $\{x_i, b_i, d_i, a_i\}$ and $\{x_i, b_i, d_i, f_i\}$. So we can conclude that there are eight codewords in each copy of $G^\times$, three codewords in each G_i and the extra codewords are among the vertices A_j, B_j. If for some j, $B_j \in C$ and $A_j \notin C$, then B_j serves as a codeword only to 1-separate itself from A_j, but this can be done by A_j, so $(C \setminus \{B_j\}) \cup \{A_j\}$ would be another optimal 1-identifying code. So we are left with the case $A_j \in C$. Then A$_j$ 1-covers one vertex x_i or $\overline{x}_i$, say x_i, and then both $C \cap V_i = \{\overline{x}_i, b_i, d_i\}$ and $C \cap V_i = \{\overline{x}_i, a_i, f_i\}$ are possible. In all cases, we have proved that there are several optimal 1-identifying codes in G^+, i.e., we have a NO answer to the constructed instance of U-OIdC$_1$. $\Diamond$

4.2.2 Extension to $r \geq 2$

As for LD codes, we do not go directly from $r = 1$ to $r \geq 2$, but start again from U-SAT, which does not change the final result; see [13, Rem. 2] about this difficulty.

Theorem 31 *Let $r \geq 2$ be any integer. There exists a polynomial reduction from U-SAT to U-IdC$_r$ and to U-OIdC$_r$: U-SAT $\rightarrow_p$ U-IdC$_r$ and U-SAT $\rightarrow_p$ U-OIdC$_r$.*

Proof. This proof is inspired by that of the *NP*-completeness of IdC$_r$ in [13].

We give a polynomial reduction starting from an instance of U-SAT, i.e., a collection $\mathcal{C}$ of m clauses over a set $\mathcal{X}$ of n variables.

In a first step, for each variable $x_i \in \mathcal{X}$, $1 \leq i \leq n$, we take the graph $G_i = (V_i, E_i)$ defined in Lemma 24. For each clause c_j, containing ε_j literals, $\varepsilon_j \geq 2$, we create two vertices, A_j and B_j, and the edge A_jB_j, and we link A_j to the ε_j vertices corresponding, in the graphs G_i, to the literals of c_j; we call these edges "membership edges". So far, we have constructed an intermediate graph, $G_I = (V_I, E_I)$.

In a second step (see Figure 11), for each membership edge and for each edge in the graphs G_i, we "paste" one copy of the graph Ω defined in

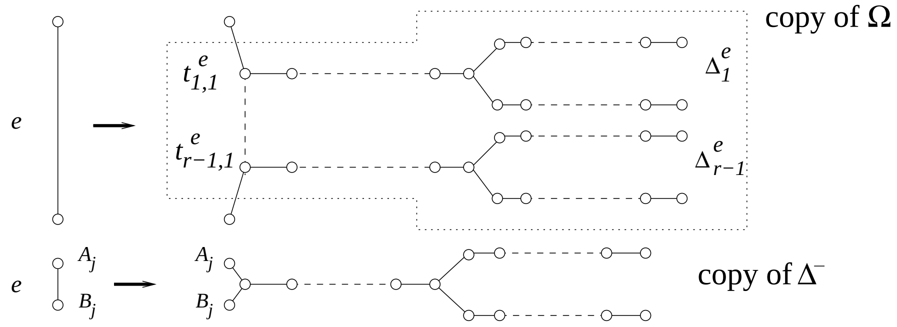

Figure 11: How the edge $e \in E_I$ is transformed in the proof of Theorem 31. If e is a membership edge or an edge in G_i, we use a copy of Ω; if $e = A_jB_j$, we use a copy of Δ^-.

Lemma 25, which is equivalent to pasting $r-1$ copies of the diapason Δ; and for each edge A_jB_j, $1 \le j \le m$, we paste *one* copy of the *shortened* diapason Δ^- defined in Corollary 26. We denote by G^+ the graph thus constructed, and set $k = 3n + (r-1)(2r+2)(|E_I| - m) + m(2r+2)$. The order of G^+ is $(6n + 2m) + (|E_I| - m)(r-1)(5r+2) + m(5r+1)$: the transformation is polynomial indeed.

The diapasons can be seen as a way of putting at distance r, in the graph G^+, the vertices in V_i, $1 \le i \le n$, and $\{A_j : 1 \le j \le m\}$ which are at distance one from one another in G_I. And so these vertices will behave with respect to each other in a way very similar to the case $r = 1$. In particular, it is still true that, in addition to codewords taken in the branches of the diapasons, at least three codewords are necessary to deal with the vertices in each V_i. Consequently, by Lemma 25(b), any optimal r-identifying code in G^+ has size at least $3n + (r-1)(2r+2)(|E_I| - m) + m(2r+2) = k$, the three terms corresponding respectively to (a) the sets V_i, $1 \le i \le n$, (b) the $r-1$ copies of the diapason on each edge which is not A_jB_j, and (c) the copy of the shortened diapason on each edge A_jB_j, $1 \le j \le m$.

The role of the copies of the shortened diapason is to r-cover A_j and B_j without r-separating them, and to r-separate A_j and B_j from the other vertices belonging to V_I.

After these introductory observations, we can conclude that, in any r-identifying code in G^+, the role of the codewords which do not belong to the branches of the diapasons, is (a) to r-separate A_j from B_j, for all $j \in \{1, \dots, m\}$; (b) to r-cover all the vertices in V_i, and to r-separate them, for all $i \in \{1, \dots, n\}$.

We claim that there is a unique solution to SAT if and only if there is a unique optimal r-identifiying code in G^+, and if and only if there is a unique r-identifying code of size at most k in G^+.

(1) Assume first that there is a unique truth assignment satisfying all the clauses. We construct the following code C: for $i \in \{1, \dots, n\}$, among the vertices $x_i \in V_i$, $\overline{x}_i \in V_i$, we put in C the vertex x_i if the literal x_i has been set TRUE, the vertex $\overline{x}_i$ if the literal x_i is FALSE, we add b_i and d_i, and we also take the unique optimal r-identifying codes in all the copies of Ω and Δ^-. Then, as in the case $r = 1$, we can check that C is an r-identifying code in G^+; in particular, for all $j \in \{1, \dots, m\}$, the vertices A_j and B_j are r-separated by C, because there is at least one true literal in the clause c_j. Also, the code C has the right size and is optimal. We can also see that, once we know that, say, $x_i \in C$, we have *no choice left* for the completion of the code, because a_i, f_i and $\overline{x}_i$ must be r-covered (the latter because no A_j is a codeword), and x_i, b_i and d_i must be r-separated by the code. So b_i and d_i necessarily are the remaining two codewords in V_i, for all $i \in \{1, \dots, n\}$.

Why is C unique? Suppose on the contrary that C^* is another r-identifying code, with $|C^*| = |C|$. Then for all $i \in \{1, \dots, n\}$, exactly three codewords take care of V_i, and C^* contains b_i, d_i and exactly one of x_i and $\overline{x}_i$. This defines a valid truth assignment for $\mathcal{X}$, by setting $x_i = \mathrm{T}$ if $x_i \in C^*$, $x_i = \mathrm{F}$ if $\overline{x}_i \in C^*$. Since $C \neq C^*$, this assignment is different from the assignment used to build C. But the fact that C^* r-separates A_j and B_j for all j shows that there is one codeword x_i or $\overline{x}_i$ r-covering A_j, which means that the clause is satisfied. Therefore, we have a second assignment satisfying the instance of SAT, a contradiction.

(2) Assume next that the answer to U-SAT is NO: this may be either because no truth assignment satisfies the instance, or because at least two assignments do; in the latter case, this would lead, using the same argument as previously, to at least two optimal r-identifying codes (of size k), and a NO answer to U-IdC$_r$ and U-OIdC$_r$. So we are left with the case when the set of clauses $\mathcal{C}$ cannot be satisfied. This implies that no r-identifying code of size k exists: we have ended the case U-IdC$_r$; next, we assume that C is an optimal r-identifying code of unknown size $|C| > k$. We know that each copy of Ω or of Δ^- contains at least $(r-1)(2r+2)$ or $2r+2$ codewords, respectively, and that each V_i requires at least three codewords. Now, where can the extra codeword(s) be?

Note that, unfortunately, Remark 16 cannot be adapted to the present construction with pasted diapasons for r-identifying codes, because in the case (b) of the Remark, when the codeword z belonging to a diapason r-separates u and v and is replaced by u or v, then u and v are not r-separated anymore. This did not matter with LD-codes, but it does for identifying codes.

Any vertex B_j is a useless codeword, because, even in the case $r = 2$, it r-covers both A_j and itself. This is also true for all the codewords that would be on the branches of any diapason (apart from the codewords $y_1, y_2, \dots y_r$, y_{2r+1} and $z_1, z_2, \dots, z_r$, z_{2r+1}), as well as for codewords on the stem of a shortened diapason (for they all r-cover both A_j and B_j).

Let us now consider the case of a codeword on the stem of a diapason pasted on an edge $e = uv$: this edge is either a membership edge or an edge in some G_i; the (only) role of this codeword is either to r-cover exactly one of u an v, and consequently to r-separate u from v, or to r-cover both u and v, and consequently to r-separate them from other vertices in V_I. We distinguish between three cases. In each case, our goal is to show that one codeword can be spared (contradiction with the optimality of C) or that several optimal codes are possible.

(i) $r = 2$. The two vertices on the stem of the unique diapason pasted on e both 2-cover u and v, so at most one of them is necessary in the code, and they are interchangeable.

(ii) $r \geq 4$. (a) All the feet of the $r - 1$ diapasons r-cover u and v, so at most one of them is necessary in the code, and they are interchangeable. (b) If, say, u is linked to $t_{1,1}$ and v to $t_{r-1,1}$, then $d_{G^+}(t_{1,r}, u) = r$, $d_{G^+}(t_{1,r-1}, u) = r - 1$, $d_{G^+}(t_{1,r}, v) = 2r - 2$, and $d_{G^+}(t_{1,r-1}, v) = 2r - 3 > r$, so there are at least two vertices on the first stem which r-cover u, not v, at most one of them is necessary in the code, and they are interchangeable. The same is true by symmetry for $t_{r-1,r}$ and $t_{r-1,r-1}$.

(iii) $r = 3$. (a) The feet of the two diapasons 3-cover u and v, and the conclusion is the same as previously. (b) Without loss of generality, we assume that we are in the following case: the codeword is $t_{1,3}$; it is the *only* vertex in Ω which 3-covers u, not v, so it 3-separates these two vertices. If $u = A_j$ for some $j \in \{1, 2, \ldots, m\}$, then it is 3-covered and 3-separated from v by the shortened diapason, and $t_{1,3}$ can be spared. So u is one of the six vertices in V_i for some $i \in \{1, 2, \ldots, n\}$. If $u \in \{x_i, \overline{x}_i, b_i, d_i\}$, that is, if u has degree at least two in G_I, then we can replace $t_{1,3}$ by another vertex suitably chosen in a diapason pasted on another edge incident to u in G_I. So we are left with the case when, say, $u = a_i$, and so $v = b_i$. But b_i is 3-covered by at least one codeword. (b1) This codeword also 3-covers a_i. Then $t_{1,3}$ can be replaced in C by a vertex 3-covering b_i, not a_i: this choice also allows to have a_i and b_i 3-covered and 3-separated by C. (b2) The codeword 3-covering b_i does not 3-cover a_i. Then $t_{1,3}$ can be replaced in C by, e.g., $t_{1,2}$, because a_i and b_i are already 3-separated by C.

So in all cases, we have at least two possible optimal codes, and we can assume from now on that each copy of Ω contains exactly $(r - 1)(2r + 2)$ codewords, and each copy of Δ^- exactly $2r + 2$ codewords.

The case when there are four (or more) codewords in a component G_i can be treated exactly like the case $r = 1$, as if the copies of Ω did not exist. This is also true if each G_i has exactly three codewords, and one extra codeword is on some A_j, because then A_j r-covers some x_i or $\overline{x}_i$. In all cases, there is more than one possibility for $V_i \cap C$.

In conclusion, whenever there are more than k codewords in an optimal code, there are several possible optimal codes, and the answer to U-OIdC$_r$

is NO. $\Diamond$

Corollary 32 *Let $r \geq 1$ be any integer. The decision problems U-IdC$_r$ and U-OIdC$_r$ are NP-hard.* $\Diamond$

Proposition 33 *Let $r \geq 2$ and $q \geq 1$ be any integers. There is a polynomial reduction from U-IdC$_{qr}$ to U-IdC$_q$ and from U-OIdC$_{qr}$ to U-OIdC$_q$: U-IdC$_{qr}$ $\to_p$ U-IdC$_q$ and U-OIdC$_{qr}$ $\to_p$ U-OIdC$_q$.*

As a particular case, we have U-IdC$_r$ $\to_p$ U-IdC$_1$ and U-OIdC$_r$ $\to_p$ U-OIdC$_1$.

Proof. See the proof of Proposition 18 and Lemma 5(b). $\Diamond$

4.2.3 An Upper Bound for the Complexity of U-IdC$_r$

Theorem 34 *There exists a polynomial reduction from U-IdC$_1$ to U-SAT: U-IdC$_1$ $\to_p$ U-SAT.*

Proof. We refer to the proof of Theorem 19, and we give here only the clauses that describe the identification problem. These clauses are constructed in the following way:

(a1) for each vertex $x^i \in V$ with neighbours $x^{n_1}, \ldots, x^{n_s}$, we take the clause of size $k(s+1)$:

$$\{x_1^i, x_2^i, \ldots, x_k^i, x_1^{n_1}, x_2^{n_1}, \ldots, x_k^{n_1}, x_1^{n_2}, \ldots, x_k^{n_2}, \ldots, x_1^{n_s}, \ldots, x_k^{n_s}\};$$

(a2) for each pair of vertices $x^i \in V$, $x^j \in V$, we consider the set $B_1(x^i) \Delta B_1(x^j) = \{x^{h_1}, x^{h_2}, \ldots, x^{h_\ell}\}$; by the assumption that G is 1-twin-free, we have $\ell > 0$. Then we simply take the clause

$$\{x_1^{h_1}, x_2^{h_1}, \ldots, x_k^{h_1}, x_1^{h_2}, \ldots, x_k^{h_2}, \ldots, x_1^{h_\ell}, \ldots, x_k^{h_\ell}\},$$

which is the second part of the clause $c_{x^i x^j}$ in the aforementioned proof. Then the argument goes exactly like for Theorem 19, in particular thanks to the characterization given by (1). $\Diamond$

By Proposition 33 or its corollary, this immediately implies that there is a polynomial reduction from U-IdC$_r$ to U-SAT.

Theorem 35 *Let $r \geq 1$ be any integer. The problem U-IdC$_r$ has complexity equivalent to that of U-SAT.*

As a consequence, U-IdC$_r$ belongs to the class DP. $\Diamond$

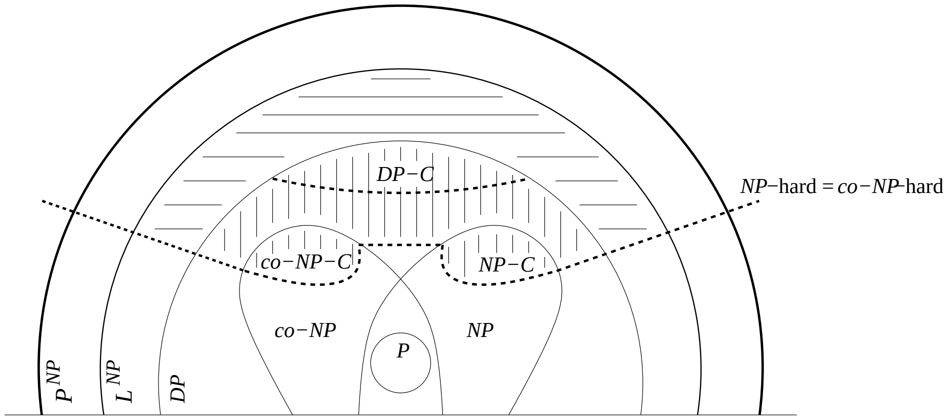

Figure 12: Some classes of complexity: Figure 1 re-visited.

4.2.4 Two Upper Bounds for the Complexity of U-OIdC$_r$

Exactly as in Section 3.2.4, we give, without proof, two results on U-OIdC$_r$; this time, Lemma 27, Corollary 28 and Proposition 29 are used.

Proposition 36 *For $r \geq 1$, the decision problem U-OIdC$_r$ belongs to the class P^{NP}. In case of a YES answer, one can give the only optimal r-identifying code within the same complexity.* ◊

Proposition 37 *For $r \geq 1$, the decision problem U-OIdC$_r$ belongs to L^{NP}.* ◊

5 Conclusion

We have established that the four decision problems U-LDC$_r$, U-IdC$_r$, U-OLDC$_r$ and U-OIdC$_r$ are, for any fixed $r \geq 1$, *NP*-hard, and that the two problems U-OLDC$_r$ and U-OIdC$_r$ belong to the class L^{NP}. For U-LDC$_r$ and U-IdC$_r$, we could go further and prove that they are *equivalent* to U-SAT and therefore belong to the class *DP*.

Conjecture Neither U-OLDC$_r$ nor U-OIdC$_r$ belong to *DP*.

Open Problem Give a better location, in the classes of complexity, for the problems U-LDC$_r$, U-IdC$_r$, U-OLDC$_r$ and U-OIdC$_r$.

We can see in Figure 12 that U-SAT, U-LDC$_r$ and U-IdC$_r$ are in the vertically hatched region, but probably not in *DP*-complete, whereas U-OLDC$_r$ and U-OIdC$_r$ are somewhere in the region that is hatched horizontally or vertically.

In [2], a characterization of the trees which admit a unique optimal 1-LD code is given.

Open Problems Extend this study to other classes of graphs; to any integer $r \geq 1$; to identifying codes. What is the complexity of the sub-problem of U-OLDC$_1$ when the instance is any tree.

In [1], the authors wonder whether

(A) U-SAT is *NP*-hard, but here we believe that what they mean is: does there exist a *polynomial* reduction from an *NP*-complete problem to U-SAT ? i.e., they use the *second* definition of *NP*-hardness;

finally, they show that (A) is true if and only if

(B) U-SAT is *DP*-complete.

So, if one is careless and considers that U-SAT is *NP*-hard without checking according to which definition, one might easily jump too hastily to the conclusion that U-SAT is *DP*-complete.

References

[1] A. Blass, Y. Gurevich, On the unique satisfiability problem, Inf. & Control 55 (1982) 80–88.

[2] M. Blidia, M. Chellali, R. Lounes, F. Maffray, Characterizations of trees with unique minimum locating-dominating sets, J. Combin. Math. Combin. Comput. 76 (2011) 225–232.

[3] C. Calabro, R. Impagliazzo, V. Kabanets, R. Paturi, The complexity of Unique k-SAT: an isolation lemma for k-CNFs, J. Comput. Syst. Sci. 74 (2008) 386–393.

[4] M. Fischermann, Block graphs with unique minimum dominating sets, Discrete Math. 240 (2001) 247–251.

[5] G. Gunther, B. Hartnell, L. R. Markus, D. Rall, Graphs with unique minimum dominating sets, Congr. Numer. 101 (1994) 55–63.

[6] C. H. Papadimitriou, On the complexity of unique solutions, J. Assoc. Comput. Machinery 31 (1984) 392–400.

[7] C. Berge, Graphes, Gauthier-Villars, Paris, 1983. English translation: Graphs, North-Holland Publishing Co., Amsterdam, 1985.

[8] R. Diestel, Graph Theory, Springer-Verlag, Berlin, 2005.

[9] 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.

[10] C. J. Colbourn, P. J. Slater, L. K. Stewart, Locating dominating sets in series parallel networks, Congr. Numer. 56 (1987) 135–162.

[11] P. J. Slater, Domination and location in graphs, National University of Singapore, Research Report No. 93, April 1983.

[12] A. Lobstein, Watching systems, identifying, locating-dominating and discriminating codes in graphs, a bibliography.
https://www.lri.fr/∼lobstein/debutBIBidetlocdom.pdf

[13] 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.

[14] G. Cohen, I. Honkala, A. Lobstein, G. Zémor, On identifying codes, Proceedings of DIMACS Workshop on Codes and Association Schemes '99, Piscataway, USA, 56 (2001) 97–109.

[15] O. Hudry, A. Lobstein, More results on the complexity of identifying problems in graphs, Theoret. Comput. Sci. 626 (2016) 1–12.

[16] O. Hudry, A. Lobstein, Some complexity considerations on the uniqueness of solutions for satisfiability and colouring problems, submitted.

[17] O. Hudry, A. Lobstein, Complexity of unique (optimal) solutions in graphs: Vertex Cover and Domination, J. Combin. Math. Combin. Comput., to appear.

[18] O. Hudry, A. Lobstein, On the complexity of determining whether there is a unique Hamiltonian cycle in a graph, J. Combin. Math. Combin. Comput., to appear.

[19] I. Honkala, O. Hudry, A. Lobstein, On the number of optimal identifying codes in a twin-free graph, Discrete Appl. Math. 180 (2015) 111–119.

[20] I. Honkala, O. Hudry, A. Lobstein, On the ensemble of optimal dominating and locating-dominating codes in a graph, Inform. Process. Lett. 115 (2015) 699–702.

[21] I. Honkala, O. Hudry, A. Lobstein, On the ensemble of optimal identifying codes in a twin-free graph, Cryptogr. Commun. – Discrete Structures, Boolean Functions & Sequences 8 (2016) 139–153.

[22] D. Auger, Identifying codes in trees and planar graphs, Electron. Notes Discrete Math. 34 (2009) 585–588.

[23] D. Auger, Minimal identifying codes in trees and planar graphs with large girth, Eur. J. Combin. 31 (2010) 1372–1384.

[24] D. Auger, I. Charon, O. Hudry, A. Lobstein, Complexity results for identifying codes in planar graphs, Internat. Trans. Oper. Res. 17 (2010) 691–710.

[25] F. Foucaud, Aspects combinatoires et algorithmiques des codes identifiants dans les graphes, Thèse de Doctorat, Université de Bordeaux 1, France, 194 pages, December 2012 (in English).

[26] F. Foucaud, S. Gravier, R. Nasesasr, A. Parreau, P. Valicov, Identifying codes in line graphs, J. Graph Theory 73 (2013) 425–448.

[27] F. Foucaud, G. B. Mertzios, R. Nasesasr, A. Parreau, P. Valicov, Identification, location-domination and metric dimension on interval and permutation graphs: II. Algorithms and complexity, Algorithmica 78 (2017) 914–944.

[28] A. Parreau, Problèmes d'identification dans les graphes, Thèse de Doctorat, Université de Grenoble, France, 214 pages, July 2012.

[29] F. Foucaud, Decision and approximation complexity for identifying codes and locating-dominating sets in restricted graph classes, J. Discrete Algorithms 31 (2015) 48–68.

[30] S. Gravier, R. Klasing, J. Moncel, Hardness results and approximation algorithms for identifying codes and locating-dominating codes in graphs, Algorithmic Oper. Res. 3 (2008) 43–50.

[31] J. Suomela, Approximability of identifying codes and locating-dominating codes, Inform. Process. Lett. 103 (2007) 28–33.

[32] C. H. Papadimitriou, M. Yannakakis, The complexity of facets (and some facets of complexity), Proceedings of the 14th Annual ACM Symposium on Theory of Computing, San Francisco, May 1982.

[33] J.-P. Barthélemy, G. D. Cohen, A. C. Lobstein, Algorithmic Complexity and Communication Problems, University College of London, London, 1996.

[34] M. R. Garey, D. S. Johnson, Computers and Intractability, a Guide to the Theory of NP-Completeness, Freeman, New York, 1979.

[35] D. S. Johnson, A catalog of complexity classes, in: van Leeuwen (Ed.), Handbook of Theoretical Computer Science, Vol. A: Algorithms and Complexity, Elsevier, 1990, Chapter 2.

[36] C. H. Papadimitriou, Computational Complexity, Addison-Wesley, Reading, 1994.

[37] S. A. Cook, The complexity of theorem-proving procedures, Proceedings of 3rd Annual ACM Symposium on Theory of Computing (1971) 151–158.

[38] T. Cormen, Algorithmic complexity, in: K. H. Rosen (Ed.), Handbook of Discrete and Combinatorial Mathematics, CRC Press, Boca Raton, 2000, pp. 1077–1085.

[39] L. Hemaspaandra, Complexity classes, in: K. H. Rosen (Ed.), Handbook of Discrete and Combinatorial Mathematics, CRC Press, Boca Raton, 2000, pp. 1085–1090.

[40] C. H. Papadimitriou, M. Yannakakis, The complexity of facets (and some facets of complexity), J. Comput. System Sci. 28 (1984) 244–259.

[41] Complexity Zoo.
https://complexityzoo.uwaterloo.ca/Complexity_Zoo