Algorithms and Complexity for Metric Dimension and Location-Domination on Interval and Permutation Graphs*

Florent Foucaud[1], George Mertzios[2]**, Reza Naserasr[3], Aline Parreau[4], and Petru Valicov[5]

[1] Université Blaise Pascal, LIMOS - CNRS UMR 6158, Clermont-Ferrand (France)
florent.foucaud@gmail.com
[2] School of Engineering and Computing Sciences, Durham University (UK)
george.mertzios@durham.ac.uk
[3] CNRS, Université Paris-Sud 11, LRI - CNRS UMR 8623, Orsay (France)
reza@lri.fr
[4] CNRS, Université de Lyon 1, LIRIS - CNRS UMR 5205 (France)
aline.parreau@univ-lyon1.fr
[5] LIF - CNRS UMR 7279, Université d'Aix-Marseille (France)
petru.valicov@lif.univ-mrs.fr

Abstract. We study the problems Locating-Dominating Set and Metric Dimension, which consist in determining a minimum-size set of vertices that distinguishes the vertices of a graph using either neighbourhoods or distances. We consider these problems when restricted to interval graphs and permutation graphs. We prove that both decision problems are NP-complete, even for graphs that are at the same time interval graphs and permutation graphs and have diameter 2. While Locating-Dominating Set parameterized by solution size is trivially fixed-parameter-tractable, it is known that Metric Dimension is $W[2]$-hard. We show that for interval graphs, this parameterization of Metric Dimension is fixed-parameter-tractable.

1 Introduction

Combinatorial identification problems have been widely studied in various contexts. The common characteristic of these problems is that we are given a combinatorial structure, and we wish to distinguish (i.e. uniquely identify) its elements by the means of a small set of selected elements. In this paper, we study two such identification problems where the instances are graphs. In the Locating-Dominating Set problem, we ask for a dominating set S such that the vertices outside of S are distinguished by their neighbourhood within S. In Metric Dimension, we wish to select a set S of vertices of a graph G such that every vertex of G is uniquely identified by its distances to the vertices of S.

* This is a short version of the full paper [16] available on arXiv:1405.2424.

** Partially supported by the EPSRC Grant EP/K022660/1.

These problems have been extensively studied since their introduction in the 1970s and 1980s. They have been applied to various areas such as network verification [2], fault-detection in networks [35], graph isomorphism testing [1] or the logical definability of graphs [25].

Important concepts and definitions. All considered graphs are finite and simple. We denote by $N[v]$, the *closed neighbourhood* of vertex v, and by $N(v)$ its *open neighbourhood*, i.e. $N[v] \setminus \{v\}$. A vertex is *universal* if it is adjacent to all the vertices of the graph. A set S of vertices of G is a *dominating set* if for every vertex v, there is a vertex x in $S \cap N[v]$. In the context of dominating sets we say that a vertex x *separates* two distinct vertices u, v if it dominates exactly one of them. Set S separates the vertices of a set X if all pairs of X are separated by a vertex of S. The distance between two vertices u, v is denoted $d(u, v)$. The following two definitions are the main concepts studied in this paper.

- (Slater [32,33]) A set L of vertices of a graph G is a *locating-dominating set* if it is a dominating set and it separates the vertices of $V(G) \setminus L$.
- (Harary and Melter [20], Slater [31]) A set R of vertices of a graph G is a *resolving set* if for each pair u, v of distinct vertices, there is a vertex x of R with $d(x, u) \neq d(x, v)$.

The smallest size of a locating-dominating set of G is the *location-domination number* of G, denoted $\gamma^{\text{LD}}(G)$. The smallest size of a resolving set of G is the *metric dimension* of G, denoted $dim(G)$. The inequality $dim(G) \leq \gamma^{\text{LD}}(G)$, relating these notions, holds for every graph G. If G has diameter 2, the two concepts are almost the same, as then, one can check that $\gamma^{\text{LD}}(G) \leq dim(G) + 1$ holds. We consider the two associated decision problems:

LOCATING-DOMINATING SET	METRIC DIMENSION
Instance: A graph G, an integer k.	*Instance:* A graph G, an integer k.
Question: Is it true that $\gamma^{\text{LD}}(G) \leq k$?	*Question:* Is it true that $dim(G) \leq k$?

We will study these problems on interval graphs and permutation graphs, which are classic graph classes that have many applications and are widely studied. They can be recognized efficiently, and many problems can be solved efficiently for graphs in these classes (see e.g. the book by Golumbic [18]). Given a set S of (geometric) objects, the *intersection graph* G of S is the graph whose vertices are associated to the elements of S and where two vertices are adjacent if and only if the corresponding elements of S intersect. Then, S is called an *intersection model* of G. An *interval graph* is the intersection graph of a set of (closed) intervals of the real line. Given two parallel lines B and T, a *permutation graph* is the intersection graph of segments of the plane which have one endpoint on B and the other endpoint on T.

Previous work. The complexity of distinguishing problems has been studied by many authors. LOCATING-DOMINATING SET was first proved to be NP-complete in [7], a result extended to bipartite graphs in [5]. This was improved to planar bipartite unit disk graphs [28] and to planar bipartite subcubic graphs [14]. LOCATING-DOMINATING SET is hard to approximate within any $o(\log n)$ factor (n is the order of the graph), with no restriction on the input graph [34]. This result was extended to bipartite graphs, split graphs and co-bipartite graphs [14]. On the positive side, Locating-Dominating Set is constant-factor approximable for bounded degree graphs [19], line graphs [14,15], interval graphs [4] and is linear-time solvable for graphs of bounded clique-width (using Courcelle's theorem [8]). Furthermore, an explicit linear-time algorithm solving Locating-Dominating Set on trees is known [32].

Metric Dimension, which has a non-local and more intricate flavour, was widely studied as well, and has (re)gained a lot of attention within the last few years. It was shown NP-complete in [17, Problem GT61]. This result has recently been extended to bipartite graphs, co-bipartite graphs, split graphs and line graphs of bipartite graphs [11], to a special subclass of unit disk graphs [23], and to planar graphs [9]. Polynomial-time algorithms for the weighted version of Metric Dimension for paths, cycles, trees, graphs of bounded cyclomatic number, cographs and partial wheels were given in [11]. A polynomial-time algorithm for outerplanar graphs was designed in [9], and one for chain graphs in [12]. It was shown in [2] that Metric Dimension is hard to approximate within any $o(\log n)$ factor for graphs of order n. This is even true for bipartite subcubic graphs, as shown in [21,22].

In light of these results, the complexity of Locating-Dominating Set and Metric Dimension for interval and permutation graphs is a natural open question (as posed in [27] and [11] for Metric Dimension on interval graphs), since these classes are standard candidates for designing efficient algorithms.

Let us say a few words about the parameterized complexity of these problems. For standard definitions and concepts in parameterized complexity, we refer to the books [10,29]. It is known that for Locating-Dominating Set, any graph of order n and solution size k satisfies $n \leq 2^k + k - 1$ [33]. Therefore, when parameterized by k, Locating-Dominating Set is trivially fixed-parameter-tractable (FPT): first check whether the above inequality holds (if not, return "no"), and if yes, use a brute-force algorithm checking all possible subsets of vertices. This is an FPT algorithm. However, Metric Dimension (again parameterized by solution size k) is W[2]-hard even for bipartite subcubic graphs [21,22]. Remarquably, the bound $n \leq D^k + k$ holds [6] (where n is the graph's order, D its diameter, and k is the size of a resolving set). Hence, for graphs of diameter bounded by a function of k, the same arguments as the previous ones yield an FPT algorithm for Metric Dimension. This holds, for example, for the class of (connected) split graphs, which have diameter at most 3. Besides this, as remarked in [22], no standard class of graphs for which Metric Dimension is FPT was previously known.

Our results. We settle the complexity of Locating-Dominating Set and Metric Dimension on interval and permutation graphs, showing that the two problems are NP-complete even for graphs that are at the same time interval graphs and permutation graphs and have diameter 2 (Section 2). Then, we present a dynamic programming algorithm (using path-decomposition) to solve Metric Dimension in FPT time on interval graphs (Section 3). Up to our knowledge, this is the first nontrivial FPT algorithm for this problem.

2 Hardness results

We will now reduce 3-DIMENSIONAL MATCHING, which is a classic NP-complete problem [24], to LOCATING-DOMINATING SET on interval graphs.

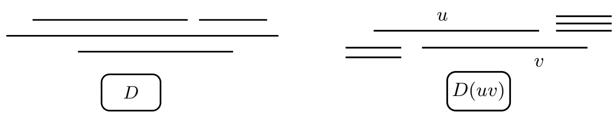
> 3-DIMENSIONAL MATCHING
> *Instance:* Three disjoint sets A, B and C each of size n, and a set $\mathcal{T}$ of m triples of $A \times B \times C$.
> *Question:* Is there a perfect 3-dimensional matching $\mathcal{M} \subseteq \mathcal{T}$ of the hypergraph $(A \cup B \cup C, \mathcal{T})$, i.e. a set of disjoint triples of $\mathcal{T}$ such that each element of $A \cup B \cup C$ belongs to exactly one of the triples?

2.1 Preliminaries and gadgets

We first define the following *dominating gadget* (a path on four vertices). The idea is to ensure that specific vertices are dominated locally, and therefore separated from the rest of the graph. We will use it extensively. The reduction is described as an interval graph, but we then show that it is also a permutation graph.

Definition 1 (Dominating gadget). *A* dominating gadget D *is a subgraph of an interval graph G inducing a path on four vertices, and such that each interval of $V(G) \setminus V(D)$ either contains all intervals of $V(D)$ or does not intersect any.*

In the following, a dominating gadget will be represented as in Figure 1(a).



(a) Dominating gadget D. (b) Choice pair u, v.

Fig. 1. Representations of dominating gadget and choice pair.

Claim 2. *If G is an interval graph containing a dominating gadget D and S is a locating-dominating set of G, then $|S \cap V(D)| \geq 2$. Moreover, there is a locating-dominating set S_D of D (called* standard solution *for D) with $|S_D| = 2$ such that no vertex of D is dominated by all the vertices of S_D. If S is an optimal locating-dominating set, then replacing $S \cap V(D)$ by the standard solution S_D, one can obtain an optimal locating-dominating set S'.*

Proof. Let $V(D) = \{x_1, x_2, x_3, x_4\}$. If $S \cap D = \emptyset$ or $S \cap D = \{x_1\}$, then x_3 and x_4 are not separated. If $S \cap D = \{x_2\}$, then x_1 and x_3 are not separated. Hence, since x_1, x_4 and x_2, x_3 are symmetric, there are at least two vertices of D in S. Moreover, the set $S_D = \{x_1, x_4\}$ is a locating-dominating set of D with the required property. $\square$

Definition 3 (Choice pair). *A pair $\{u, v\}$ of intervals is called a* choice pair *if u, v both contain the intervals of a common dominating gadget (denoted $D(uv)$), and such that none of u, v contains the other.*

See Figure 1(b) for an illustration of a choice pair. Intuitively, a choice pair gives us the choice of separating it from the left or from the right: since none of u, v is included in the other, the intervals intersecting u but not v can only be located at one side of u; the same holds for v. In our construction, we will make sure that, except for the choice pairs, all pairs of intervals will be easily separated using domination gadgets. Our aim will then be to separate the choice pairs. We have the following claim that follows directly from Claim 2:

Claim 4. *Let S be a locating-dominating set of a graph G and $\{u, v\}$ be a choice pair in G. If the solution $S \cap V(D(uv))$ for the dominating gadget $D(uv)$ is the standard solution S_D, both vertices u and v are dominated, separated from all vertices in $D(uv)$ and from all vertices not intersecting $D(uv)$.*

We now define the central gadget of the reduction, the *transmitter gadget.* Roughly speaking, it allows to transmit information across an interval graph.

Definition 5 (Transmitter gadget). *Let P be a set of two or three choice pairs in an interval graph G. A* transmitter gadget *$Tr(P)$ is a subgraph of G consisting of a path on seven vertices $\{u, uv^1, uv^2, v, vw^1, vw^2, w\}$ and five dominating gadgets $D(u)$, $D(uv)$, $D(v)$, $D(vw)$, $D(w)$ such that the following properties are satisfied:*
- *u and w are the only vertices of $Tr(P)$ that separate the pairs of P.*
- *The intervals of the dominating gadget $D(u)$ (resp. $D(v)$, $D(w)$) are included in interval u (resp. v, w) and no interval of $Tr(P)$ other than u (resp. v, w) intersects $D(u)$ (resp. $D(v)$, $D(w)$).*
- *Pair $\{uv^1, uv^2\}$ is a choice pair and no interval of $V(Tr(P)) \setminus (D(uv^1, uv^2) \cup \{uv^1, uv^2\})$ intersects both intervals of the pair. The same holds for pair $\{vw^1, vw^2\}$.*
- *The choice pairs $\{uv^1, uv^2\}$ and $\{vw^1, vw^2\}$ cannot be separated by intervals of G other than u, v and w.*

Figure 2 illustrates a transmitter gadget and shows the succinct graphical representation that we will use. As shown in the figure, we may use a "box" to denote $T_r(P)$. This box does not include the choice pairs of P but indicates where they are situated. Note that the middle pair $\{y_1, y_2\}$ could also be separated (from the left) by u instead of w, or it may not exist at all.

The following claim shows how transmitter gadgets will be used in the main reduction.

Claim 6. *Let G be an interval graph with a transmitter gadget $Tr(P)$ and let S be a locating-dominating set of G. We have $|S \cap Tr(P)| \geq 11$ and if $|S \cap Tr(P)| = 11$, no pair of P is separated by a vertex in $S \cap Tr(P)$. Moreover, there exist two sets of vertices of $Tr(P)$, $S^-_{Tr(P)}$ and $S^+_{Tr(P)}$ of size 11 and 12 respectively, such*

that the following holds:
• *The set $S^{-}_{Tr(P)}$ dominates all the vertices of $Tr(P)$ and separates all the pairs of $Tr(P)$ but no pairs in P.*
• *The set $S^{+}_{Tr(P)}$ dominates all the vertices of $Tr(P)$, separates all the pairs of $Tr(P)$ and all the pairs in P.*

Proof. By Claim 2, we must have $|S \cap Tr(P)| \geq 10$ with 10 vertices of S belonging to the dominating gadgets. In order that uv^1, uv^2 are separated, at least one vertex of $\{u, uv^1, uv^2, v\}$ belongs to S (recall that by definition the intervals not in $Tr(P)$ cannot separate the choice pairs in $Tr(P)$), and similarly, for the choice pair $\{vw^1, vw^2\}$, at least one vertex of $\{v, vw^1, vw^2, w\}$ belongs to S. Hence $|S \cap Tr(P)| \geq 11$ and if $|S \cap Tr(P)| = 11$, vertex v must be in S and neither u nor w are in S. Therefore, no pair of P is separated by a vertex in $S \cap Tr(P)$.

We now prove the second part of the claim. Let S_{dom} be the union of the five standard solutions S_D of the dominating gadgets of $Tr(P)$. Let $S^{-}_{Tr(P)} = S_{dom} \cup \{v\}$ and $S^{+}_{Tr(P)} = S_{dom} \cup \{u, w\}$. The set S_{dom} has 10 vertices and so $S^{-}_{Tr(P)}$ and $S^{+}_{Tr(P)}$ have respectively 11 and 12 vertices. Each interval of $Tr(P)$ either contains a dominating gadget or is part of a dominating gadget and is therefore dominated by a vertex in S_{dom}. Hence, pairs of vertices that are not intersecting the same dominating gadget are clearly separated. By Claim 2, no vertex in a dominating gadget D is dominated by all vertices of S_D, hence a vertex adjacent to the whole of D is separated from all the vertices of D. Also, by Claim 2, all pairs of vertices inside a dominating gadget are separated by S_{dom}. Therefore, the only remaining pairs to consider are the choice pairs. Note that they are separated both at the same time either by v or by $\{u, w\}$. Hence the two sets $S^{-}_{Tr(P)}$ and $S^{+}_{Tr(P)}$ are both dominating and separating the vertices of $Tr(P)$. Moreover, since $S^{+}_{Tr(P)}$ contains u and w, it also separates the pairs of P. □

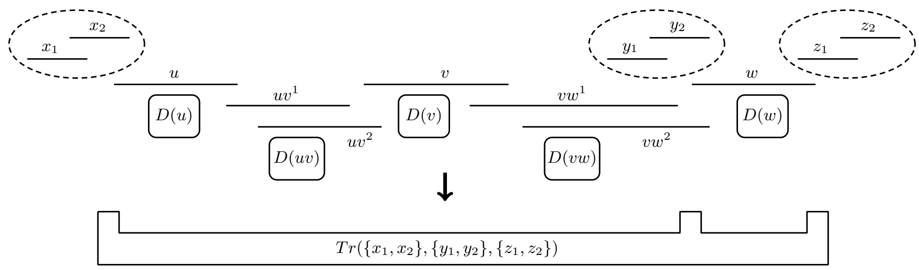


Fig. 2. Transmitter gadget $Tr(\{x_1, x_2\}, \{y_1, y_2\}, \{z_1, z_2\})$ and its "box" representation.

2.2 The main reduction

We now describe the reduction. Each element $x \in A \cup B \cup C$ is modelled by a choice pair $\{f_x, g_x\}$. Each triple of $\mathcal{T}$ is modelled by a triple gadget:

Definition 7 (Triple gadget). *Let $T = \{a, b, c\}$ be a triple of $\mathcal{T}$. The* triple gadget *$G_t(T)$ is an interval graph consisting of four choice pairs $p = \{p_1, p_2\}$, $q = \{q_1, q_2\}$, $r = \{r_1, r_2\}$, $s = \{s_1, s_2\}$ together with their associated dominating gadgets $D(p)$, $D(q)$, $D(r)$, $D(s)$ and five transmitter gadgets $Tr(p, q)$, $Tr(r, s)$, $Tr(s, a)$, $Tr(p, r, b)$ and $Tr(q, r, c)$, where:*
- *$a = \{f_a, g_a\}$, $b = \{f_b, g_b\}$ and $c = \{f_c, g_c\}$;*
- *Except for the choice pairs, for each pair of intervals of $G_t(T)$, its two intervals intersect different subsets of $\{D(p), D(q), D(r), D(s)\}$;*
- *In each transmitter gadget $Tr(P)$ and for each choice pair $\pi \in P$, the intervals of π intersect the same intervals except for the vertices u, v, w of $Tr(P)$;*
- *The intervals of $V(G) \setminus V(G_t(T))$ that are intersecting only a part of the gadget intersect accordingly to the transmitter gadget definition and do not separate the choice pairs p, q, r and s.*

An illustration of a triple gadget is given in Figure 3. We remark that p, q, r and s in $G_t(\{a, b, c\})$, are all functions of $\{a, b, c\}$ but to simplify the notations we simply write p, q, r and s.

We will call the sets $S^-_{Tr(P)}$ and $S^+_{Tr(P)}$ the *tight* and *non-tight standard solutions* of $Tr(P)$.

Claim 8. *Let G be a graph with a triple gadget $G_t(T)$ and S be a locating-dominating set of G. We have $|S \cap G_t(T)| \geq 65$ and if $|S \cap G_t(T)| = 65$, no choice pair corresponding to a, b or c is separated by a vertex in $S \cap G_t(T)$. Moreover, there exist two sets of vertices of $G_t(T)$, $S^-_{G_t(T)}$ and $S^+_{G_t(T)}$ of size 65 and 66 respectively, such that the following holds.*
- *The set $S^-_{G_t(T)}$ dominates all the vertices of $G_t(T)$ and separates all the pairs of $G_t(T)$ but does not separate any choice pairs corresponding to $\{a, b, c\}$.*
- *The set $S^+_{G_t(T)}$ dominates all the vertices of $G_t(T)$, separates all the pairs of $G_t(T)$ and separates the choice pairs corresponding to $\{a, b, c\}$.*

Proof. The proof is similar of the proof of Claim 6. Each transmitter gadget must contain at least 11 vertices of S, and each of the four dominating gadgets of the choice pairs p,q,r,s must contain 2 vertices of S. Hence there must be already 63 vertices of $G_t(T)$ in S. Furthermore, to separate the choice pair s, $Tr(r, s)$ or $Tr(s, a)$ must be non-tight (since this choice pair cannot be separated by other vertices of the graph) and in the same way, to separate the choice pair p, $Tr(p, q)$ or $Tr(p, r, b)$ must be non-tight. Then at least two transmitter gadgets are non-tight and we have $|S \cap G_t(T)| \geq 65$. If $|S \cap G_t(T)| = 65$, exactly two transmitter gadgets are non-tight and they can only be $Tr(r, s)$ and $Tr(p, q)$. Hence the choice pairs corresponding to $\{a, b, c\}$ are not separated by the vertices of $G_t(T) \cap S$.

For the second part of the claim, the set $S^-_{G_t(T)}$ is defined by taking the union of the tight standard solutions of $Tr(s,a)$, $Tr(q,r,c)$ and $Tr(p,r,b)$), the non-tight standard solutions of $Tr(p,q)$ and $Tr(r,s)$ and the standard solutions of the dominating gadgets $D(p)$, $D(q)$, $D(r)$ and $D(s)$. The set $S^+_{G_t(T)}$ is defined by taking the union of the non-tight standard solutions of $Tr(s,a)$, $Tr(q,r,c)$ and $Tr(p,r,b)$, the tight standard solutions of $Tr(p,q)$ and $Tr(r,s)$ and the standard solutions of the dominating gadgets $D(p)$, $D(q)$, $D(r)$ and $D(s)$. By Claim 6, the definition of a dominating gadget and the fact that the only intervals sharing the same sets of dominating gadgets are the choice pairs, all intervals of $G_t(T)$ are dominated and all the pairs of intervals except the choice pairs are separated by both $S^-_{G_t(T)}$ and $S^+_{G_t(T)}$. The choice pairs p, q, r and s are separated by the non-tight solutions of the transmitter gadgets. Hence $S^-_{G_t(T)}$ and $S^+_{G_t(T)}$ are dominating and separating all the intervals of $G_t(T)$.

When S contains $S^-_{G_t(T)}$, the transmitter gadgets $Tr(s,a)$, $Tr(q,r,c)$ and $Tr(p,r,b)$ are tight. Hence $S^-_{G_t(T)}$ does not separate any choice pairs among $\{a,b,c\}$. On the other hand, since $S^+_{G_t(T)}$ contains the non-tight solution of $Tr(s,a)$, $Tr(q,r,c)$ and $Tr(p,r,b)$, the three choice pairs $\{a,b,c\}$ are separated by $S^+_{G_t(T)}$. □

A triple gadget with 65 vertices (resp. 66) in the solution is said to be *tight* (resp. *non-tight*). We call the sets $S^-_{G_t(T)}$ and $S^+_{G_t(T)}$ the *tight* and *non-tight standard solutions* of $G_t(T)$.

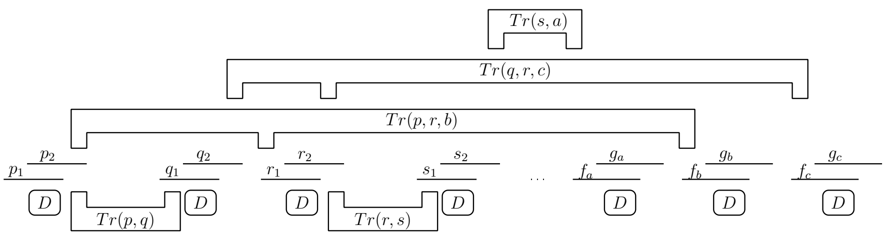


Fig. 3. Triple gadget $G_t(T)$ with $T = \{a,b,c\}$ together with the choice pairs of elements a, b and c. We recall that these choice pairs are not part of $G_t(T)$.

Given an instance $(A,B,C,\mathcal{T})$ of 3-DIMENSIONAL MATCHING with $|A| = |B| = |C| = n$ and $|\mathcal{T}| = m$, we construct the interval graph $G = G(A,B,C,\mathcal{T})$ as follows.

• As mentioned previously, to each element x of $A \cup B \cup C$, we assign a distinct choice pair $\{f_x, g_x\}$ in G. The intervals of any two distinct choice pairs $\{f_x, g_x\}, \{f_y, g_y\}$ are disjoint and they are all in $\mathbb{R}^+$.

• For each triple $T = \{a,b,c\}$ of $\mathcal{T}$ we first associate an interval I_T in $\mathbb{R}^-$ such that for any two triples T_1 and T_2, I_{T_1} and I_{T_2} do not intersect. Then inside I_T, we build the choice pairs $\{p_1,p_2\}$, $\{q_1,q_2\}$, $\{r_1,r_2\}$, $\{s_1,s_2\}$. Finally, using the choice pairs already associated to elements a, b and c we complete this to a triple gadget.

• When placing the remaining intervals of the triple gadgets, we must ensure that triple gadgets do not "interfere": for every dominating gadget D, no interval in $V(G)\setminus V(D)$ must have an endpoint inside D. Similarly, the choice pairs of each triple gadget or transmitter gadget must only be separated by intervals among u, v and w of its corresponding private transmitter gadget. For intervals of distinct triple gadgets, this is easily done by our placement of the triple gadgets. To ensure that the intervals of transmitter gadgets of the same triple gadget do not interfere, we proceed as follows. We place the whole gadget $Tr(p,q)$ inside interval u of $Tr(p,r,b)$. Similarly, the whole $Tr(r,s)$ is placed inside interval v of $Tr(p,r,b)$ and w of $Tr(q,r,c)$. One has to be more careful when placing the intervals of $Tr(p,r,b)$ and $Tr(q,r,c)$. In $Tr(p,r,b)$, we must have that interval u separates p from the right of p. We also place u so that it separates r from the left of r. Intervals uv^1, uv^2 both start in r_1, so that u also separates uv^1, uv^2 without these intervals interfering with the ones of r. Intervals uv^1, uv^2 continue until after pair s. In $Tr(q,r,c)$, we place u so that it separates q from the right, and we place w so that it separates r from the right; intervals uv^1, uv^2, v lie strictly between q and r; intervals vw^1, vw^2 intersect r_1, r_2 but stop before the end of r_2 (so that w can separate both pairs vw^1, vw^2 and r but without these pairs interfering). It is now easy to place $Tr(s,a)$ between s and a.

The graph $G(A,B,C,\mathcal{T})$ has $159m+18n$ vertices and the interval representation described by our procedure can be obtained in polynomial time. We are now ready to state the main result of this section.

Theorem 9. *$(A,B,C,\mathcal{T})$ has a perfect 3-dimensional matching if and only if $G = G(A,B,C,\mathcal{T})$ has a locating-dominating set with $65m+7n$ vertices.*

Proof. Let $\mathcal{M}$ be a perfect 3-dimensional matching of $(A,B,C,\mathcal{T})$. Let S^+ (resp. S^-) be the union of all the non-tight (resp. tight) standard solutions $S^+_{G_t(T)}$ for $T\in\mathcal{M}$ (resp. $S^-_{G_t(T)}$ for $T\notin\mathcal{M}$). Let S_{dom} be the union of all the standard solutions of the dominating gadgets corresponding to the choice pairs of the elements. Then $S = S^+\cup S^-\cup S_{dom}$ is a locating-dominating set of size $65m+7n$. Indeed, by the definition of the dominating gadgets, all the intervals inside a dominating gadget are dominating and separating from all the other intervals. All the other intervals intersect at least one dominating gadget and thus are dominated. Furthermore, two intervals that are not a choice pair do not intersect the same set of dominating gadgets and thus are separated by one of the dominating gadgets. Finally, the choice pairs inside a triple gadget are separated by Claim 8 and the choice pairs corresponding to the elements of $A\cup B\cup C$ are separated by the non-tight standard solutions of the triple gadgets corresponding to the perfect matching.

Now, let S be a locating-dominating set of size $65m+7n$. By Claims 2 and 8, we can assume that the solution is standard on all triple gadgets and on the dominating gadgets. Let n_2 be the number of non-tight triple gadgets in S. By Claim 8, there must by at least $65m+n_2$ vertices of S inside the m triple gadgets and $6n$ vertices of S for the dominating gadgets of the $3n$ elements of $A\cup B\cup C$.

Hence $65m+n_2+6n \leq 65m+7n$ and we have $n_2 \leq n$. Each non-tight triple gadget can separate three choice pairs corresponding to the elements of $A \cup B \cup C$. Hence, if $n_2 < n$, it means that at least $3(n-n_2)$ choice pairs corresponding to elements are not separated by a triple gadget. By the separation property, the only way to separate a choice pair $\{f_x, g_x\}$ without using a non-tight triple gadget is to have f_x or g_x in S. Hence we need $3(n-n_2)$ vertices to separate these $3(n-n_2)$ choice pairs, and these vertices are neither in the triple gadgets nor in the dominating gadgets. Hence S has size at least $65m + n_2 + 3(n - n_2) > 65m + 7n$, leading to a contradiction.

Therefore $n_2 = n$, there are exactly n non-tight triple gadgets, each of them separates 3 choice element pairs and since there are $3n$ elements, the non-tight triple gadgets separate distinct choice pairs. Hence the set of triples $\mathcal{M}$ corresponding to the non-tight triple gadgets is a perfect 3-dimensional matching of $(A, B, C, \mathcal{T})$. □

Theorem 9 shows that Locating-Dominating Set is NP-complete for interval graphs. In fact, the constructed graph is also a permutation graph.

Proposition 10. *The graph $G = G(A, B, C, \mathcal{T})$ is a permutation graph.*

Proof. We represent the permutation graph using its intersection model of segments as defined in the introduction. A dominating gadget will be represented as in Figure 4, and it is clearly a permutation graph. The transmitter gadget is also a permutation graph, see Figure 5 for an illustration. Now using these two representations it is easy to complete the permutation diagram of the whole construction. □

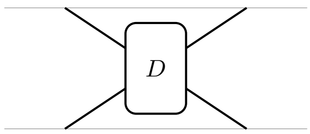


Fig. 4. Representation of dominating gadget as permutation diagram intersection model.

Corollary 11. Locating-Dominating Set *is NP-complete for graphs that are both interval and permutation graphs.*

2.3 Diameter 2 and consequence for Metric Dimension

We now describe a self-reduction for Locating-Dominating Set for graphs with a universal vertex (hence, graphs of diameter 2), and a similar reduction from Locating-Dominating Set to Metric Dimension.

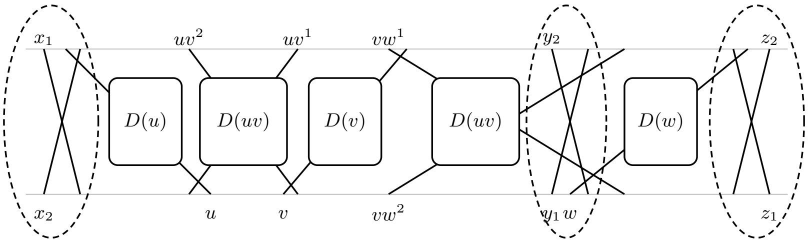


Fig. 5. Permutation diagram intersection model of a transmitter gadget.

Let G be a graph. Let $f_1(G)$ be the graph obtained from G by adding a universal vertex u and a neighbour v of u of degree 1. Let $f_2(G)$ be the graph obtained from G by adding two adjacent universal vertices u, u' and two non-adjacent vertices v and w that are only adjacent to u and u'. See Figure 6 for an illustration. One can show that $\gamma^{\text{LD}}(f_1(G)) = \gamma^{\text{LD}}(G) + 1$, and $dim(f_2(G)) = \gamma^{\text{LD}}(G) + 2$.

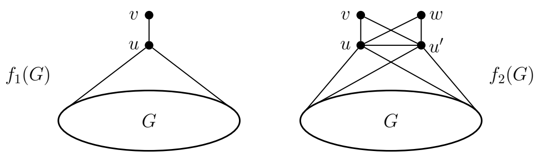


Fig. 6. Two reductions for diameter 2.

Theorem 12. *Let $\mathcal{C}$ be a class of graphs that is closed under the graph transformation f_1. If* Locating-Dominating Set *is NP-complete for graphs in $\mathcal{C}$, then it is also NP-complete for graphs in $\mathcal{C}$ that have diameter* 2.

Proof. It is enough to prove that $\gamma^{\text{LD}}(f_1(G)) = \gamma^{\text{LD}}(G) + 1$. Let S be a locating-dominating set of G. Then $S' = S \cup \{v\}$ is also a locating-dominating set of $f_1(G)$: all vertices within $V(G) \setminus S'$ are distinguished by S as they were in G; vertex u is the only vertex dominated by the whole set $S \cup \{v\}$. Hence, $\gamma^{\text{LD}}(f_1(G)) \leq \gamma^{\text{LD}}(G) + 1$.

It remains to prove the converse. Let S_1 be a locating-dominating set of $f_1(G)$. Observe that $|S_1 \cap \{u, v\}| \geq 1$ since v must be dominated. Hence if $S_1 \setminus \{u, v\}$ is a locating-dominating set of G, we are done. Let us assume the contrary. Then, necessarily $u \in S_1$ since v does not dominate any vertex of $V(G)$. But u is a universal vertex, hence u does not separate any pair of vertices of $V(G)$. Therefore, $S_1 \setminus \{u\}$ separates all pairs, but does not dominate some vertex $x \in V(G)$: we have $N[x] \cap S_1 = \{u\}$. Note that x is the only such vertex of G. This implies that $v \in S_1$ (otherwise x and v are not separated by S_1). But then $(S_1 \setminus \{u, v\}) \cup \{x\}$ is a locating-dominating set of G of size $|S_1| - 1$. This completes the proof. $\square$

Theorem 13. *Let $\mathcal{C}$ be a class of graphs that is closed under the graph transformation f_2. If* Locating-Dominating Set *is NP-complete for graphs in $\mathcal{C}$, then* Metric Dimension *is also NP-complete for graphs in $\mathcal{C}$ that have diameter* 2.

Proof. It is enough to prove that $dim(f_2(G)) = \gamma^{\text{LD}}(G)+2$. Let S be a locating-dominating set of G. We claim that $S_2 = S \cup \{u,v\}$ is a resolving set of $f_2(G)$. Every vertex of S_2 is clearly distinguished. Every original vertex of G is determined by a distinct set of vertices of S that are at distance 1 of it. Vertex u' is the only vertex to be at distance 1 of each vertex in S_2. Finally, vertex w is the only vertex to be at distance 1 of u and at distance 2 from all other vertices of S_2.

For the other direction, assume B is a resolving set of $f_2(G)$. Then necessarily one of u,u' (say u) belongs to B; similarly, one of v,w (say v) belongs to B. Hence, if the restriction $B_G = B \cap V(G)$ is a locating-dominating set of G, we are done. Otherwise, since no vertex among u,u',v,w may distinguish any pair of G and since vertices of G are at distance at most 2 in $f_2(G)$, all the sets $N[x] \cap B$ are distinct for $x \in V(G) \setminus B_G$. But B_G is not a locating-dominating set, so there is a (unique) x vertex of G that is not dominated by B_G in G. If $|B \cap \{u,u',v,w\}| \geq 3$, $B_G \cup \{x\}$ is a locating-dominating set of size at most $|B|-2$ and we are done. Otherwise, note that in $f_2(G)$, x is at distance 1 from u and at distance 2 from all other vertices of B. But this is also the case for w, which is not separated from x by B, which is a contradiction. □

Since Theorems 12 and 13 can be applied to interval graphs and permutation graphs, Corollary 11 implies the following.

Corollary 14. Locating-Dominating Set *and* Metric Dimension *are NP-complete for diameter* 2*-graphs that are both interval and permutation graphs.*

3 Metric Dimension is FPT on interval graphs

We now prove that Metric Dimension (parameterized by solution size) is FPT on interval graphs. The algorithm is based on dynamic programming over a path-decomposition.

Given an interval graph G, we can assume that in its interval model, all endpoints are distinct, and that the intervals are closed. We define two natural total orderings of $V(G)$ based on this model: $x <_L y$ if and only if the left endpoint of x is smaller then the left endpoint of y, and $x <_R y$ if and only if the right endpoint of x is smaller than the right endpoint of y. We will work with the fourth distance-power G^4 of the input graph G. One can show (see Proposition 15) that G^4 is an interval graph and has an interval model inducing the same orders $<_L$ and $<_R$ as G. Our algorithm will use dynamic programming on a *nice path-decomposition* of G^4.

3.1 Preliminaries

We start by stating a few properties and lemmas that are necessary for our algorithm.

Interval graphs Given an interval graph G, we can assume that in its interval model, all endpoints are distinct, and that the intervals are closed intervals. Given an interval I, we will denote by $\ell(I)$ and by $r(I)$ its left and right endpoints, respectively. We define two natural total orderings of $V(G)$ based on this model: $x <_L y$ if and only if the left endpoint of x is smaller then the left endpoint of y, and $x <_R y$ if and only if the right endpoint of x is smaller than the right endpoint of y.

Given a graph G, its *distance-power* G^d is the graph obtained from G by adding an edge between each pair of vertices at distance at most d in G. Raychaudhuri [30] proved that for any integer $d \geq 2$, the distance-power G^d of an interval graph G is also an interval graph (see also Flotow [13]). In fact, this can be strengthened as follows.

Proposition 15. *Let G be an interval graph with an interval model inducing orders $<_L$ and $<_R$, and let $d \geq 2$ be an integer. Then the power graph G^d is an interval graph with an interval model inducing the same orders $<_L$ and $<_R$ as G (that can be computed in polynomial time).*

Proof. Consider an interval model $\mathcal{I}$ of G, which can be computed in polynomial time (see [3]). We will construct (in polynomial time) an interval model $\mathcal{I}_d$ of G^d that induces the same orders $<_L$ and $<_R$ as $\mathcal{I}$ for G.

Let x be an interval of $\mathcal{I}$. Let u_x be the last interval of $<_L$ that is at distance at most d of x. Let $r_d(x)$ be a point located after $\ell(u_x)$ and before the next left endpoint. If for two intervals x and y we have $u_x = u_y$ and if $x <_R y$, then we choose $r_d(x)$ and $r_d(y)$ such that $r_d(x) < r_d(y)$. For each interval x of $\mathcal{I}$, we put the interval $x_d = [\ell(x), r_d(x)]$ in $\mathcal{I}_d$. In other words, all the intervals of $\mathcal{I}$ are extended to the right until being adjacent to the last interval at distance d, and locally preserving the order $<_R$.

Since the left endpoints are the same in $\mathcal{I}$ and $\mathcal{I}_d$, the left order induced by $\mathcal{I}_d$ is clearly $<_L$. Let $x <_R y$ in $\mathcal{I}$. Let u_x (resp. u_y) be the last interval of $<_L$ at distance d of x (in G). If $u_x = u_y$, then we have $r_d(x) < r_d(y)$. Otherwise, since x is finishing before y, we necessarily have $u_x <_L u_y$, and thus $r_d(x) < r_d(y)$. In both cases, the right order is respected.

We now prove $\mathcal{I}_d$ is an interval model of G^d. Let $x <_L y$ be two intervals of $\mathcal{I}$ and x_d and y_d the corresponding intervals in $\mathcal{I}_d$. We prove that x_d and y_d are intersecting in $\mathcal{I}_d$ if and only if there are at distance at most d in G. Assume first that $d(x,y) \leq d$ (in G). By definition of $r_d(x)$, we have $\ell(y) < r_d(x)$. Thus x_d and y_d are intersecting in $\mathcal{I}_d$. For the converse, assume that x_d and y_d are adjacent. Since $x_d <_L y_d$, x_d is finishing after the beginning of y_d in $\mathcal{I}_d$, so $r_d(x) > \ell(y)$. By construction, the last interval u_x of $<_L$ at distance at most d of x is either y or starting after y. Thus $x <_L y \leq_L u_x$ and, in G, $d(x,y) \leq d(x,u_x) \leq d$. □

Tree-decompositions

Definition 16. *A* tree-decomposition *of a graph G is a pair $(\mathscr{T}, \mathcal{X})$, where $\mathscr{T}$ is a tree and $\mathcal{X} := \{X_t : t \in V(\mathscr{T})\}$ is a collection of subsets of $V(G)$ (called*

bags*), and they must satisfy the following conditions:*
(i) $\bigcup_{t \in V(\mathscr{T})} X_t = V(G)$*;*
(ii) for every edge $uv \in E(G)$*, there is a bag of* $\mathcal{X}$ *that contains both* u *and* v*;*
(iii) for every vertex $v \in V(G)$*, the set of bags containing* v *induces a connected subtree of* $\mathscr{T}$*.*

Given a tree-decomposition of $(\mathscr{T}, \mathcal{X})$, the maximum size of a bag X_t over all tree nodes t of $\mathscr{T}$ minus one is called the *width* of $(\mathscr{T}, \mathcal{X})$. The minimum width of a tree-decomposition of G is the *treewidth* of G. The notion of tree-decomposition has been used extensively in algorithm design, especially via dynamic programming over the tree-decomposition.

We consider a *rooted* tree-decomposition by fixing a root of $\mathscr{T}$ and orienting the tree edges from the root toward the leaves. A rooted tree-decomposition is *nice* (see Kloks [26]) if each node t of $\mathscr{T}$ has at most two children and falls into one of the four types:
(i) *Join* node: t has exactly two children t_1 and t_2, and $X_t = X_{t_1} = X_{t_2}$.
(ii) *Introduce* node: t has a unique child t', and $X_t = X_{t'} \cup \{v\}$.
(iii) *Forget* node: t has a unique child t', and $X_t = X_{t'} \setminus \{v\}$.
(iv) *Leaf* node: t is a leaf node in $\mathscr{T}$.

Given a tree-decomposition, a nice tree-decomposition of the same width always exists and can be computed in linear time [26].

If G is an interval graph, we can construct a tree-decomposition of G (in fact, a path-decomposition) with special properties.

Proposition 17. *Let* G *be an interval graph with clique number* ω *and an interval model inducing orders* $<_L$ *and* $<_R$*. Then,* G *has a nice tree-decomposition* $(\mathscr{P}, \mathcal{X})$ *of width* $\omega - 1$ *that can be computed in linear time, where moreover:*
(a) $\mathscr{P}$ *is a path (hence there are no join nodes);*
(b) every bag is a clique;
(c) going through $\mathscr{P}$ *from the leaf to the root, the order in which vertices are introduced in an introduce node corresponds to* $<_L$*;*
(d) going through $\mathscr{P}$ *from the leaf to the root, the order in which vertices are forgotten in a forget node corresponds to* $<_R$*;*
(e) the root's bag is empty, and the leaf's bag contains only one vertex.

Proof. Given a graph G, one can decide if it is an interval graph and, if so, compute a representation of it in linear time [3]. This also gives us the ordered set of endpoints of intervals of G.

To obtain $(\mathscr{P}, \mathcal{X})$, we first create the leaf node t, whose bag X_t contains the interval with smallest left endpoint. We then go through the set of all endpoints of intervals of G, from the second smallest to the largest. Let t be the last created node. If the new endpoint is a left endpoint $\ell(I)$, we create an introduce node t' with $X_{t'} = X_t \cup \{I\}$. If the new endpoint is a right endpoint $r(I)$, we create a forget node t' with $X_{t'} = X_t \setminus \{I\}$. In the end we create the root node as a forget node t with $X_t = \emptyset$ that forgets the last interval of G.

Observe that one can associate to every node t (except the root) a point p of the real line, such that the bag X_t contains precisely the set of intervals containing p: if t is an introduce node, p is the point $\ell(I)$ associated to the creation of t, and if t is a forget node, it is the point $r(I)+\epsilon$, where ϵ is sufficiently small and $r(I)$ is the endpoint associated to the creation of t. This set forms a clique, proving Property (b). Furthermore this implies that the maximum size of a bag is ω, hence the width is at most $\omega - 1$ (and at least $\omega - 1$ since every clique must be included in some bag).

Moreover it is clear that the procedure is linear-time, and by construction, Properties (a), (c), (d), (e) are fulfilled.

Let us now show that $(\mathscr{P}, \mathcal{X})$ is a tree-decomposition. It is clear that every vertex belongs to some bag, proving Property (i) of Definition 16. Moreover let u, v be two adjacent vertices of G, and assume $u <_L v$. Then, consider the introduce node of $\mathscr{P}$ where v is introduced. Since u has started before v but has not stopped before the start of v, both u, v belong to X_t, proving Property (ii). Finally, note that a vertex v appears exactly in all bags starting from the bag where v is introduced, until the bag where v is forgotten. Hence Property (iii) is fulfilled, and the proof is complete. □

The following lemma immediately follows from Proposition 15.

Lemma 18. *Let G be an interval graph with an interval model inducing orders $<_L$ and $<_R$, let $d \geq 1$ be an integer and let $(\mathscr{P}, \mathcal{X})$ be a tree-decomposition of G^d obtained by Proposition 17 (recall that by Proposition 15, G^d is an interval graph, and it has an intersection model inducing the same orders $<_L$ and $<_R$). Then the following holds.*
(a) Let t be an introduce node of $(\mathscr{P}, \mathcal{X})$ with child t', with $X_t = X_{t'} \cup \{v\}$. Then, X_t contains every vertex w in G such that $d_G(v, w) \leq d$ and $w <_L v$.
(b) Let t' be the child of a forget node t of $(\mathscr{P}, \mathcal{X})$, with $X_t = X_{t'} \setminus \{v\}$. Then, $X_{t'}$ contains every vertex w in G such that $d_G(v, w) \leq d$ and $v <_R w$.

Proof. We prove (a), the proof of (b) is the same. By Proposition 15, we may assume that $<_L$ is the same in G and G^d. By construction of $(\mathscr{P}, \mathcal{X})$ the introduce node of v contains all intervals w of G^d intersecting v with $w <_L v$ in G^d. Hence $w <_L v$ in G as well, and $d_G(v, w) \leq d$. □

Lemmas for the algorithm We now prove a few preliminary results necessary for the argumentation. We first start with a definition and a series of lemmas based on the linear structure of an interval graph, that will enable us to defer the decision-taking (about which vertex should belong to the solution in order distinguish a specific vertex pair) to later steps of the dynamic programming.

Definition 19. *Given a vertex u of an interval graph G, the* rightmost path *$P_R(u)$ of u is the path $u_0^R, \ldots, u_p^R$ where $u = u_0^R$, for every u_i^R ($i \in \{0, \ldots, p-1\}$) u_{i+1}^R is the neighbour of u_i^R with the largest right endpoint, and thus u_p^R is the interval in G with largest right endpoint. Similarly, we define the* leftmost path *$P_L(u) = u_0^L, \ldots, u_q^L$ where for every u_i^L ($i \in \{0, \ldots, q-1\}$) u_{i+1}^L is the neighbour of u_i^L with the* smallest left *endpoint.*

Note that $P_R(u)$ and $P_L(u)$ are two shortest paths from u_0^R to u_p^R and u_q^L, respectively.

Lemma 20. *Let u be an interval in an interval graph G, and let v be an interval starting after the end of u_{i-1}^r, where $u_{i-1}^r \in P_R(u)$. Then $d(u,v) = d(u_i^R, v) + i$. Similarly, if v ends before the start of an interval u_i^{l-1} in $P_L(u)$, then $d(u,v) = d(u_i^L, v) + i$.*

Proof. We prove the claim only for the first case, the second one is symmetric. Consider the shortest path from u to v by choosing the interval intersecting u that has the largest right endpoint, and iterating. This path coincides with $P_R(u)$ until it contains some interval u_j^R such that u_j^R intersects v. Since v starts after the end of u_{i-1}^r, we have $i \leq j$. Thus, the interval u_i^R lies on a shortest path from u to v, and hence $d(u,v) = d(u_i^R, v) + d(u, u_i^R) = d(u_i^R, v) + i$. □

Lemma 21. *Let u, v be a pair of intervals of an interval graph G. For every $u_i^R \in P_R(u)$ and $v_i^R \in P_R(v)$, we have $d(u_i^R, v_i^R) \leq d(u,v)$.*

Proof. First note that, by letting $w = u_i^R$, we have $w_1^R = u_{i+1}^R$. Therefore, we only need to prove the claim for $i = 1$.

If u and v are adjacent, then either $v = u_1^R$ (then we are done) or u_1^R must end after v. Then, either u_1^R intersects v_1^R, or $u_1^R = v_1^R$. In both cases, $d(u_1^R, v_1^R) \leq 1$.

If u and v are not adjacent, we can assume that u ends before v starts. Then, by Lemma 20, $d(u_1^R, v) = d(u,v) - 1$ and $d(u_1^R, v_1^R) \leq d(u_1^R, v) + d(v, v_1^R) = d(u,v) - 1 + 1 = d(u,v)$. □

We say that a pair u, v of intervals in an interval graph G is separated by interval x *strictly from the right* (*strictly from the left*, respectively) if x starts after both right endpoints of u, v (ends before both left endpoints of u, v respectively). In other words, x is not a neighbour of any of u and v.

The next lemma is crucial for our algorithm.

Lemma 22. *Let u, v, x be three intervals in an interval graph G and let i be an integer such that x starts after both right endpoints of $u_i^R \in P_R(u)$ and $v_i^R \in P_R(v)$. Then the three following facts are equivalent:*
(1) x separates u_i^R, v_i^R;
(2) for every j with $0 \leq j \leq i$, x separates u_j^R, v_j^R;
(3) for some j with $0 \leq j \leq i$, x separates u_j^R, v_j^R.
Similarly, assume that x ends before both left endpoints of $u_i^L \in P_L(u)$ and $v_i^L \in P_L(v)$. Then the three following facts are equivalent:
(i) x separates u_i^L, v_i^L;
(ii) for every j with $0 \leq j \leq i$, x separates u_j^L, v_j^L;
(iii) for some j with $0 \leq j \leq i$, x separates u_j^L, v_j^L.

Proof. We prove only (1)–(3), the proof of (i)–(iii) is symmetric. Let $0 \leq j \leq i$ and $u' = u_j^R$ and $v' = v_j^R$. Then $(u')_{i-j}^R = u_i^R$ and $(v')_{i-j}^R = v_i^R$. By Lemma 20,

$d(u_j^R, x) = d(u_i^R, x) + (j - i)$ and similarly $d(v_j^R, x) = d(v_i^R, x) + (j - i)$. Hence x separates u_i^R and v_i^R if and only if it separates u_j^R and v_j^R which implies the lemma. □

We now introduce a local version of resolving sets that will be used in our algorithm.

Definition 23. *A* distance-2 resolving set *is a set S of vertices where for each pair u, v of vertices at distance at most 2, there is a vertex $x \in S$ such that $d(u, s) \neq d(v, s)$.*

Using the following lemma, we can manage to "localize" the dynamic programming, as we will only need to distinguish pairs of vertices that will be present together in one bag.

Lemma 24. *Any distance-2 resolving set of an interval graph G is a resolving set of G.*

Proof. Assume to the contrary that S is a distance-2 resolving set of an interval graph G but not a resolving set. It means that there is a pair of vertices u, v at distance at least 3 that are not separated by any vertex of S. Among all such pairs, we choose one, say $\{u, v\}$, such that $d(u, v)$ is minimized. Without loss of generality, we assume that u ends before v starts.

Consider u_1^R (v_1^L, respectively), the interval intersecting u (v, respectively) that has the largest right endpoint (smallest left endpoint, respectively). We have $u_1^R \neq v_1^L$ (since $d(u, v) \geq 3$) and $d(u_1^R, v_1^L) = d(u, v) - 2 < d(u, v)$. By minimality, u_1^R and v_1^L are separated by some vertex $s \in S$. But s does not separate u and v, thus $s \notin \{u_1^R, v_1^L\}$.

Without loss of generality, we can assume that $d(u_1^R, s) < d(v_1^L, s)$. In particular, $d(v_1^L, s) \geq 2$ and s is ending before v_1^L starts. Thus, by Lemma 20, $d(v, s) = d(v_1^L, s) + 1$. However, we also have $d(u, s) \leq d(u_1^R, s) + 1 \leq d(v_1^L, s) < d(v, s)$. Hence s is separating u and v, a contradiction. □

The next lemma enables us to upper-bound the size of the bags in our tree-decompositions, which will induce diameter 4-subgraphs of G.

Lemma 25. *Let G be an interval graph with a resolving set of size k, and let $B \subseteq V(G)$ be a subset of vertices such that for each pair $u, v \in B$, $d_G(u, v) \leq d$. Then $|B| \leq 4dk^2 + (2d + 3)k + 1$.*

Proof. Let $s_1, \ldots, s_k$ be the elements of a resolving set S of size k in G. Consider an interval representation of G, and let $\mathcal{B}$ be the minimal segment of the real line containing all intervals corresponding to vertices of B.

For each i in $\{1, \ldots, k\}$, consider the leftmost and rightmost paths $P_L(s_i)$ and $P_R(s_i)$, as defined in Definition 19. Let L^i be the ordered set of left endpoints of intervals of $P_L(s_i)$, and let R^i be the ordered set of right endpoints of intervals of $P_R(s_i)$. Note that intervals at distance j of s_i in G are exactly the intervals finishing between $\ell(u_{j+1}^L)$ and $\ell(u_j^L)$, or starting between $r(u_j^R)$ and $r(u_{j+1}^R)$.

Hence, for any interval of G, its distance to s_i is uniquely determined by the position of its right endpoint in the ordered set L^i and the position of its left endpoint in the ordered set R^i. Moreover, note that, since any two vertices in B are at distance at most d, $\mathcal{B}$ may contain at most d points of L^i and at most d points of R^i.

Therefore, $\mathcal{B}$ may contain at most $2kd$ points of $\bigcup_{1\leq i\leq k}(L^i \cup R^i)$. This set of points defines a natural partition $\mathcal{P}$ of $\mathcal{B}$ into at most $2kd+1$ sub-segments, and any interval of B is uniquely determined by the positions of its two endpoints in $\mathcal{P}$ (if two intervals start and end in the same part of $\mathcal{P}$, they are not separated by S, a contradiction).

Let $I \in B \setminus S$. For a fixed $i \in \{1,\dots,k\}$, by definition of the sets L^i, the interval I cannot contain two points of L^i and similarly, it cannot contain two points of R^i. Thus, I contains at most $2k$ points of the union of all the sets L^i and R^i. Therefore, if P denotes a part of $\mathcal{P}$, there are at most $2k+1$ intervals with left endpoints in P. In total, there are at most $(2kd+1)\cdot(2k+1)$ intervals in $B \setminus S$ and hence $|B| \leq (2kd+1)\cdot(2k+1)+k = 4dk^2+(2d+3)k+1$. □

3.2 The algorithm

We are now ready to describe our algorithm.

Theorem 26. Metric Dimension *can be solved in time $2^{O(k^4)}n$ on interval graphs, i.e. it is FPT on this class when parameterized by the solution size k.*

Proof. Let $(\mathscr{P},\mathcal{X})$ be a path-decomposition of the interval graph G^4 obtained using Proposition 17. Our algorithm is a dynamic programming over $(\mathscr{P},\mathcal{X})$.

Let $(\mathscr{P},\mathcal{X})$ be a path-decomposition of G^4 (which by Proposition 15 is an interval graph) obtained using Proposition 17.

The algorithm is a bottom-up dynamic programming on $(\mathscr{P},\mathcal{X})$. By Proposition 17(b), every bag of $(\mathscr{P},\mathcal{X})$ is a clique of G^4 (i.e. a subgraph of diameter at most 4 in G) and hence by Lemma 25, it has $O(k^2)$ vertices. Thanks to Lemma 22, we can "localize" the problem by considering for separation, only pairs of vertices present together in the current bag. Let us now be more precise.

For a node t in $\mathscr{P}$, we denote by $\mathcal{P}(X_t)$ the pairs of intervals in X_t that are at distance at most 2 (in G).

For each node t, we compute a set of *configurations* using the configurations of the child of t in $\mathscr{P}$. A configuration contains full information about the local solution on X_t, but also stores necessary information about the vertex pairs that still need to be separated. More precisely, a configuration $C = (\mathsf{S}, \mathsf{sep}, \mathsf{toSepR}, \mathsf{cnt})$ of t is a tuple where:

- $\mathsf{S} \subseteq X_t$ contains the vertices of the sought solution belonging to X_t;
- $\mathsf{sep} : \mathcal{P}(X_t) \to \{0,1,2\}$ assigns, to every pair in $\mathcal{P}(X_t)$, value 0 if the pair has not yet been separated, value 2 if it has been separated strictly from the left, and value 1 otherwise;

– $\mathsf{toSepR} : \mathcal{P}(X_t) \to \{0, 1\}$ assigns, to every pair in $\mathcal{P}(X_t)$, value 1 if the pair needs to be separated strictly from the right (and it is not yet separated), and value 0 otherwise;
– cnt is an integer counting the total number of vertices in the partial solution that has led to C.

Starting with the leaf of $\mathscr{P}$, for each node our algorithm goes through all possibilities of choosing S; however, sep, toSepR and cnt are computed along the way. At each new visited node t of $\mathscr{P}$, a set of configurations is constructed from the configuration sets of the child of t. The algorithm makes sure that all the information is consistent, and that configurations that will not lead to a valid resolving set (or with $\mathsf{cnt} > k$) are discarded.

Leaf node: For the leaf node t, since by Proposition 17(e) $X_t = \{v\}$, we create two configurations $C_1 = (\emptyset, \mathsf{sep}, \mathsf{toSepR}, 0)$ and $C_2 = (\{v\}, \mathsf{sep}, \mathsf{toSepR}, 1)$ (where sep and toSepR are empty in both configurations).

Introduce node: Let t be an introduce node with t' its child, where $X_t = X_{t'} \cup \{v\}$. For every configuration $(\mathsf{S}', \mathsf{sep}', \mathsf{toSepR}', \mathsf{cnt}')$ of t', we create two configurations $C_1 = (\mathsf{S}' \cup \{v\}, \mathsf{sep}_1, \mathsf{toSepR}_1, \mathsf{cnt}' + 1)$ (corresponding to the case where v is in the partial solution) and $C_2 = (\mathsf{S}', \mathsf{sep}_2, \mathsf{toSepR}_2, \mathsf{cnt}')$ (where v is not added to the partial solution).

The elements of sep_1 and toSepR_1 in C_1 are first copied from sep' and toSepR', and updated by checking, for every pair x, y of $\mathcal{P}(X_t)$ whether v separates x, y (note that v cannot separate any such pair strictly from the left). Also note that v is separated from all other vertices since it belongs to the solution, but for $x = v$ we still need to check whether v, y are strictly separated from the left (in which case we set $\mathsf{sep}_1(v, y) = 2$, otherwise $\mathsf{sep}_1(v, y) = 1$). To do this, we compute v_1^L and y_1^L (by Lemma 18(a) they both belong to X_t), and we first check if they are strictly separated from the left, which is true if and only if $\mathsf{sep}'(v_1^L, y_1^L) = 2$. If v_1^L and y_1^L are separated strictly from the left, then so are v and y. Otherwise, if v and y are still strictly separated from the left, there must be an interval z ending before the left endpoint of y and separating v, y. Since z does not separate v_1^L and y_1^L strictly from the left, z must be adjacent to y_1^L and thus $d_G(v, z) \leq 4$ (since $d_G(v, y) \leq 2$). Then, by Lemma 18, z belongs to X_t, thus it is enough to test whether any vertex of S' separates v, y strictly from the left. Moreover, we let $\mathsf{toSepR}_1(v, y) = 0$.

For C_2, we must compute $\mathsf{sep}_2(v, w)$ and $\mathsf{toSepR}_2(v, w)$ for every w such that $(v, w) \in \mathcal{P}(X_t)$. To do so, we consider the first intervals of $P_L(v)$ and $P_L(w)$. We let $\mathsf{sep}_2(v, w) = 2$ if for the pair v_1^L, w_1^L with $v_1^L \in P_L(v)$ and $w_1^L \in P_L(w)$, $\mathsf{sep}'(v_1^L, w_1^L) = 2$, or if some vertex of S' separates v, w strictly from the left. Otherwise, if v, w are separated by a neighbour of w, we set $\mathsf{sep}_2(v, w) = 1$. We also compute toSepR_2 from toSepR' by letting $\mathsf{toSepR}_2(v, w) = 0$ and copying all other values.

If $\mathsf{cnt} + 1 > k$, C_1 is discarded. The remaining valid configurations among C_1, C_2 are added to the set of configurations of t. If in this set, there are two configurations that differ only on their value of cnt, we only keep the one with the smallest value of cnt.

Forget node: Let t be a forget node and t' be its child, with $X_t = X_{t'} \setminus \{v\}$. For every configuration $(\mathsf{S}', \mathsf{sep}', \mathsf{toSepR}', \mathsf{cnt}')$ of t', we create the configuration $(\mathsf{S}' \setminus \{v\}, \mathsf{sep}, \mathsf{toSepR}, \mathsf{cnt}')$. We create sep and toSepR by copying all entries $\mathsf{sep}'(x,y)$ and $\mathsf{toSepR}'(x,y)$ such that $x, y \in \mathcal{P}(X_t)$.

For every vertex w in X_t such that $d_G(v,w) \leq 2$, if $\mathsf{sep}'(v,w) = 0$ or $\mathsf{toSepR}'(v,w) = 1$ (i.e. v,w still need to be separated strictly from the right), we determine v_1^R and w_1^R and let $\mathsf{toSepR}(v_1^R, w_1^R) = 1$ (note that $d_G(v, v_1^R) = 1$, $d_G(v, w_1^R) \leq 3$, $v <_R v_1^R$ and $v <_R w_1^R$, hence by Lemma 18(b) $v_1^R, w_1^R \in X_{t'}$ and hence $v_1^R, w_1^R \in X_t$). However, if $v_1^R = w_1^R$, we discard the current configuration. Indeed, by Lemma 22, v,w cannot be separated strictly from the right: any shortest path to any of v,w from some vertex x whose interval starts after both right endpoints of v,w must go through $v_1^R = w_1^R$ and hence $d(x, v_1^R) = d(x, w_1^R)$. We also discard the configuration if v_1^R or w_1^R does not exist (i.e. v or w is the rightmost interval of G).

Finally, if there are two configurations that differ only on their value of cnt, again we only keep the one with the smallest value of cnt.

Root node: At root node t, since by Proposition 17(e) $X_t = \emptyset$, t has at most one configuration. We output "yes" only if this configuration exists, and if $\mathsf{cnt} \leq k$. Otherwise, we output "no".

We now analyze the algorithm.

Correctness. We claim that G has a resolving set of size at most k if and only if the root node of $\mathscr{P}$ contains a valid configuration. By Lemma 24, this is equivalent to proving that G has an optimal *distance*-2 *resolving set* of size at most k if and only if the root node of $\mathscr{P}$ contains a valid configuration. First, assume that the dynamic programming has succeeded, i.e. the root bag contains a valid configuration C. Assume that C has smallest value cnt. We want to prove that the union of all partial solutions S of all configurations that have led to the computation of C is a valid optimal solution S.

We first prove that for every pair u,v of vertices with $d_G(u,v) \leq 2$ and $u <_R v$, S separates u,v. By Lemma 18(b), u,v are present together in the child t' of forget node t of $\mathscr{P}$ where u is forgotten. Let $C_{t'} = (\mathsf{S}', \mathsf{sep}', \mathsf{toSepR}', \mathsf{cnt}')$ and $C_t = (\mathsf{S}, \mathsf{sep}, \mathsf{toSepR}, \mathsf{cnt})$ be the configurations of t', t that have led to the end configuration C. In the computation of C_t, since C_t was not discarded, we either had $\mathsf{sep}'(u,v) > 0$ in $C_{t'}$ or the algorithm has set $\mathsf{toSepR}(u_r^1, v_r^1) = 1$, in which case $u_1^R \neq v_1^R$. Assume we had $\mathsf{sep}'(u,v) = 1$. Then, in some configuration $C_{t''}$ that has led to computing $C_{t'}$ (possibly $t' = t''$), u and v were separated by some vertex in S belonging to $C_{t''}$, and we are done. If $\mathsf{sep}'(u,v) = 2$, similarly either u,v have been separated by some vertex of S belonging to a (possibly earlier) configuration, or we had $\mathsf{sep}(u_i^L, v_i^L) = 2$, in which case by Lemma 22 we are also done. If however, the algorithm has set $\mathsf{toSepR}(u_1^R, v_1^R) = 1$, recall that unless in some bag u_1^R, v_1^R is separated strictly from the right, when we forget u_1^R we set $\mathsf{toSepR}(u_2^R, v_2^R) = 1$. Hence, since C was a valid configuration (and has not been discarded), at some step we have separated u_i^R, v_i^R strictly from the right, which by Lemma 22 implies that u, v are separated by S, and we are done.

Moreover S is optimal because we have chosen C so as to minimize the size cnt of the overall solution. At each step, the algorithm discards, among equivalent configurations, the ones with larger values of cnt, ensuring that the size of the solution is minimized. This proves our claim.

For the converse, assume that G has an optimal distance-2 resolving set S of size at most k. We will need the following claim.

Claim. Let u, v be a pair of vertices with $d_G(u, v) \leq 2$. Then, any vertex x that could separate u, v neither strictly from the right nor strictly from the left is present in some bag together with both u, v.

Proof of claim. Necessarily, x is a neighbour of one of u, v in G. Hence $d_G(x, u) \leq 3$ and $d_G(x, v) \leq 3$. If $x <_L v$, by Lemma 18(a) x, u, v are present in the bag where v is introduced. If $v <_L x$, similarly x, u, v are present in the bag where x is introduced. □

(◇)

We will prove that some configuration C was computed using a series of configurations where for each node t of $\mathscr{P}$, the right subset $S \cap X_t$ was guessed. By contradiction, if this was not the case, then at some step of the algorithm we would have discarded a configuration C' although it arised from guessing the correct partial solution of S. Since S is optimal, C' was not discarded because there was a copy of C' with different value of counter cnt (otherwise this copy would lead to a solution strictly smaller than S). Hence the discarding of C' has happened at a node t that is a forget node. Assume that t is a forget node where vertex v was forgotten (assume t' is the child of t in $\mathscr{P}$). This happens only if for some $w \in X_t$ with $d_G(v, w) \leq 2$, we had either (i) $\mathsf{sep}'(v, w) = 0$ and $v_1^R = w_1^R$, or (ii) $\mathsf{toSepR}(v, w) = 1$ and $v_1^R = w_1^R$. If (i) holds, then v, w are considered not to be separated, although they are actually separated (by our assumption on C'). Since $v_1^R = w_1^R$, v_1^R and w_1^R cannot be separated strictly from the right, hence by Lemma 22 v, w are not separated strictly from the right. If they are not separated strictly from the left, Claim 3.2 implies a contradiction because the vertex separating v, w was present together in a bag with v, w and hence we must have $\mathsf{sep}'(v, w) = 1$. Hence, v, w are separated strictly from the left. But again by Lemma 22, this means that some vertices v_i^L, w_i^L in $P_R(v) \times P_R(w)$ have been separated strictly from the left (assume that i is maximal with this property). Since by Lemma 21, $d_G(v_i^L, w_i^L) \leq 2$, by Lemma 18 these two vertices were present in some bag simultaneously, together with the vertex that is strictly separating them from the left (and has distance at most 4 from w_i^L). Then in the configuration corresponding to this bag, $\mathsf{sep}(v_i^L, w_i^L) = 2$, and we had $\mathsf{sep}'(v, w) = 2$ in C', a contradiction. If (ii) holds, there exists a pair x, y such that in some earlier configuration, we had $\mathsf{sep}(x, y) = 0$, $v = x_i^R \in P_R(x)$ and $w = y_i^R \in P_R(y)$. By the same reasoning as for (i) we obtain a contradiction. This proves this side of the implication, and completes the proof of correctness.

Running time. At each step of the dynamic programming, we compute the configurations of a bag from the set of configurations of the child bag. The computation of each configuration is polynomial in the size of the current bag of $(\mathscr{P}, \mathcal{X})$. Since a configuration is precisely determined by a tuple $(\mathsf{S}, \mathsf{sep}, \mathsf{toSepR})$ (if there are two configurations where only cnt differs, we only keep the one with smallest value), there are at most $2^{|X_t|}3^{|X_t|^2}2^{|X_t|^2} \leq 3^{2|X_t|^2}$ configurations for a bag X_t. Hence, in total the running time is upper-bounded by $2^{O(b^2)}n$, where b is the maximum size of a bag in $(\mathscr{P}, \mathcal{X})$. Since any bag induces a subgraph of G of diameter at most 4, by Lemma 25, $b = O(k^2)$. Therefore $2^{O(b^2)}n = 2^{O(k^4)}n$, as claimed. □

4 Conclusion

We proved that both Locating-Dominating Set and Metric Dimension are NP-complete even for diameter 2-graphs that are both interval and permutation graphs. This is in contrast to related problems such as Dominating Set, which is linear-time solvable on both classes. However, we do not know their complexity for unit interval graphs or bipartite permutation graphs (note that both problems are polynomial-time solvable on chain graphs, a subclass of bipartite permutation graphs [12]). We also note that our reduction can be adapted to related problems such as Identifying Code (see the full version of this paper [16]).

Regarding our FPT algorithm for Metric Dimension on interval graphs, we do not know whether the result holds for graph classes such as permutation graphs or chordal graphs. The main obstacles for adapting our algorithm to chordal graphs are (i) that Lemma 22, which is essential for our algorithm, heavily relies on the two orderings induced by intersection models of interval graphs, and (ii) that Lemma 24 is not true for chordal graphs. Indeed, Figure 7 shows a family of arbitrarily large graphs with a distance-2 resolving set (black vertices) that is not a resolving set (u and v are not separated).

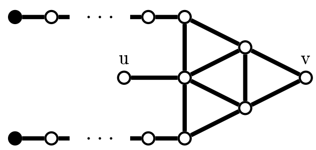


Fig. 7. A family of chordal graphs with a distance-2 resolving set that is not resolving.

Acknowledgments. We thank Adrian Kosowski for helpful discussions.

References

1. L. Babai. On the complexity of canonical labelling of strongly regular graphs. *SIAM J. Comput.* 9(1):212–216, 1980.

2. Z. Beerliova, F. Eberhard, T. Erlebach, A. Hall, M. Hoffmann, M. Mihalák, L. S. Ram. Network discovery and verification. *IEEE J. Sel. Area Comm.* 24(12):2168–2181, 2006.
3. K. S. Booth and G. S. Lueker. Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms. *J. Comput. Syst. Sci.* 13(3):335–379, 1976.
4. N. Bousquet, A. Lagoutte, Z. Li, A. Parreau, S. Thomassé. Identifying codes in hereditary classes of graphs and VC-dimension. Manuscript, 2014. arXiv:1407.5833
5. I. Charon, O. Hudry, A. Lobstein. Minimizing the size of an identifying or locating-dominating code in a graph is NP-hard. *Theor. Comput. Sci.* 290(3):2109–2120, 2003.
6. G. Chartrand, L. Eroh, M. Johnson, O. Oellermann. Resolvability in graphs and the metric dimension of a graph. *Disc. Appl. Math.* 105(1-3):99–113, 2000.
7. C. Colbourn, P. J. Slater, L. K. Stewart. Locating-dominating sets in series-parallel networks. *Congr. Numer.* 56:135–162, 1987.
8. B. Courcelle. The monadic second-order logic of graphs. I. Recognizable sets of finite graphs. *Inf. Comput.* 85(1):12–75, 1990.
9. J. Diaz, O. Pottonen, M. Serna, E. Jan van Leeuwen. On the complexity of metric dimension. *Proc. ESA 2012*, LNCS 7501:419–430, 2012.
10. R. G. Downey, M. R. Fellows. *Fundamentals of Parameterized Complexity.* Springer, 2013.
11. L. Epstein, A. Levin, G. J. Woeginger. The (weighted) metric dimension of graphs: hard and easy cases. *Algorithmica*, to appear.
12. H. Fernau, P. Heggernes, P. van't Hof, D. Meister, R. Saei. Computing the metric dimension for chain graphs. *Inform. Process. Lett.* 115:671–676, 2015.
13. C. Flotow. On powers of m-trapezoid graphs. *Disc. Appl. Math.* 63(2):187–192, 1995.
14. F. Foucaud. Decision and approximation complexity for identifying codes and locating-dominating sets in restricted graph classes. *J. Discrete Alg.* 31:48–68, 2015.
15. F. Foucaud, S. Gravier, R. Naserasr, A. Parreau, P. Valicov. Identifying codes in line graphs. *J. Graph Theor.* 73(4):425–448, 2013.
16. F. Foucaud, G. Mertzios, R. Naserasr, A. Parreau, P. Valicov. Identification, location-domination and metric dimension on interval and permutation graphs. II. Algorithms and complexity. Manuscript, 2014. arXiv:1405.2424
17. M. R. Garey, D. S. Johnson. *Computers and intractability: a guide to the theory of NP-completeness*, W. H. Freeman, 1979.
18. M. C. Golumbic. *Algorithmic graph theory and perfect graphs*, Elsevier, 2004.
19. S. Gravier, R. Klasing, J. Moncel. Hardness results and approximation algorithms for identifying codes and locating-dominating codes in graphs. *Algorithmic Oper. Res.* 3(1):43–50, 2008.
20. F. Harary, R. A. Melter. On the metric dimension of a graph. *Ars Comb.* 2:191–195, 1976.
21. S. Hartung. *Exploring parameter spaces in coping with computational intractability.* PhD Thesis, TU Berlin, Germany, 2014.
22. S. Hartung, A. Nichterlein. On the parameterized and approximation hardness of metric dimension. *Proc. CCC 2013*:266–276, 2013.
23. S. Hoffmann, E. Wanke. Metric dimension for Gabriel unit disk graphs is NP-Complete. *Proc. ALGOSENSORS 2012*:90-92, 2012.
24. R. M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

25. J. H. Kim, O. Pikhurko, J. Spencer, O. Verbitsky. How complex are random graphs in First Order logic? *Random Struct. Alg.* 26(1-2):119–145, 2005.
26. T. Kloks. *Treewidth, Computations and Approximations.* Springer, 1994.
27. P. Manuel, B. Rajan, I. Rajasingh, Chris Monica M. On minimum metric dimension of honeycomb networks. *J. Discrete Alg.* 6(1):20–27, 2008.
28. T. Müller, J.-S. Sereni. Identifying and locating-dominating codes in (random) geometric networks. *Comb. Probab. Comput.* 18(6):925–952, 2009.
29. R. Niedermeier. *Invitation to Fixed-Parameter Algorithms.* Oxford University Press, 2006.
30. A. Raychaudhuri. On powers of interval graphs and unit interval graphs. *Congr. Numer.* 59:235–242, 1987.
31. P. J. Slater. Leaves of trees. *Congr. Numer.* 14:549–559, 1975.
32. P. J. Slater. Domination and location in acyclic graphs. *Networks* 17(1):55–64, 1987.
33. P. J. Slater. Dominating and reference sets in a graph. *J. of Math. and Phys. Sci.* 22(4):445–455, 1988.
34. J. Suomela. Approximability of identifying codes and locating-dominating codes. *Inform. Process. Lett.* 103(1):28–33, 2007.
35. R. Ungrangsi, A. Trachtenberg, D. Starobinski. An implementation of indoor location detection systems based on identifying codes. *Proc. INTELLCOMM 2004*, LNCS 3283:175–189, 2004.