On the Complexity of Metric Dimension*

Josep Díaz[1], Olli Pottonen[1], Maria Serna[1], and Erik Jan van Leeuwen[2]

[1] Departament de Llenguatges i Sistemes Informatics, UPC, Barcelona, Spain.
diaz,mjserna@lsi.upc.edu, olli.pottonen@iki.fi
[2] Dept. Computer, Control, Managm. Eng., Sapienza University of Rome, Italy
E.J.van.Leeuwen@dis.uniroma1.it

Abstract. The metric dimension of a graph G is the size of a smallest subset $L \subseteq V(G)$ such that for any $x, y \in V(G)$ there is a $z \in L$ such that the graph distance between x and z differs from the graph distance between y and z. Even though this notion has been part of the literature for almost 40 years, the computational complexity of determining the metric dimension of a graph is still very unclear. Essentially, we only know the problem to be NP-hard for general graphs, to be polynomial-time solvable on trees, and to have a $\log n$-approximation algorithm for general graphs. In this paper, we show tight complexity boundaries for the Metric Dimension problem. We achieve this by giving two complementary results. First, we show that the Metric Dimension problem on bounded-degree planar graphs is NP-complete. Then, we give a polynomial-time algorithm for determining the metric dimension of outerplanar graphs.

1 Introduction

Given a graph $G = (V, E)$, its *metric dimension* is the cardinality of a smallest set $L \subseteq V$ such that for every pair $x, y \in V$, there is a $z \in L$ such that the length of a shortest path from z to x is different from the length of a shortest path from z to y. In this case we say that x, y are *resolved* by z and L. Elements of the set L are called *landmarks*. A set $L \subseteq V$ that resolves all pairs of vertices is called a *resolving set*. The problem of finding the metric dimension of a given graph G is called Metric Dimension, but is also known as *Harary's problem*, and the *locating number* or *rigidity* problem. The problem was defined independently by Harary and Melter [15] and Slater [21].

The reasons for studying this problem are three-fold. First, even though the problem is part of Garey and Johnson's famous book on computational intractability [14], very little is known about the computational complexity of this problem. Garey and Johnson proved thirty years ago that the decision version of Metric Dimension is NP-complete on general graphs [18] (another proof appears in [19]). It was shown that there exists a $2 \log n$-approximation algorithm

* J. Díaz and M. Serna are partially supported by TIN-2007-66523 (FORMALISM) and SGR 2009-2015 (ALBCOM). O. Pottonen was supported by the Finnish Cultural Foundation. E.J. van Leeuwen is supported by ERC StG project PAAl no. 259515.

on arbitrary graphs [19], which is best possible within a constant factor under reasonable complexity assumptions [2, 16]. Hauptmann et al. [16] show hardness of approximation on sparse graphs and on complements of sparse graphs. On the positive side, fifteen years ago, Khuller et al. [19] gave a linear-time algorithm to compute the metric dimension of a tree, as well as characterizations for graphs with metric dimension 1 and 2. Epstein et al. [10] provide hardness results for split graphs, bipartite graphs, co-bipartite graphs, and line graphs of bipartite graphs, together with polynomial-time algorithms for a weighted variant of metric dimension for several simple graphs including paths, trees, and cycles. To the best of our knowledge, no further results are known about the complexity of this problem. It is thus interesting if the substantial, long-standing gap on the tractability of this problem (between trees and general graphs) can be closed.

Second, the problem has received a lot of attention from researchers in different disciplines and it is frequently studied from an analytical point of view (see e.g. [1, 16] and references therein). For example, a recent survey by Bailey and Cameron [1] notes an interesting connection to group theory and graph isomorphism. It was also shown to be applicable to certain cop-and-robber games [11]. Therefore it makes sense to reopen the investigation on the computational complexity of Metric Dimension and close the above-mentioned complexity gap.

The third reason for studying Metric Dimension, particularly on (outer) planar graphs, is that known techniques in the area do not seem to apply to it. Crucially, it seems difficult to formulate the problem as an MSOL-formula in a natural way, implying that Courcelle's Theorem [5] does not apply. Hence there is no easy way to show that the problem is polynomial-time solvable on graphs of bounded treewidth. Also, the line of research pioneered by Baker [3], which culminated in the recent meta-theorems on planar graphs using the framework of bidimensionality [8, 13], does not apply, as Metric Dimension does not exhibit the required behavior. For example, the metric dimension of a grid is two [19], whereas bidimensionality requires it to be roughly linear in the size of the grid. Moreover, the problem is not closed under contraction. This behavior of Metric Dimension contrasts that of many other problems, even of nonlocal problems such as Feedback Vertex Set. Hence by studying the Metric Dimension problem, there is an opportunity to extend the toolkit that is available to us on planar graphs.

Our Results. In the present work, we significantly narrow the tractability gap of Metric Dimension. From the hardness side, we show that Metric Dimension on planar graphs, called Planar Metric Dimension, is NP-hard, even for bounded-degree planar graphs. From the algorithmic side, we show that there is a polynomial-time algorithm to find the metric dimension of outerplanar graphs, significantly improving the known algorithm on trees.

The crux of both of these results is our ability to deal with the fact that the Metric Dimension problem is extremely *nonlocal*. In particular, a landmark can resolve vertices that are very far away from it. The paper thus focusses on constraining the effects of a landmark to a small area. The NP-hardness proof does this by constructing a specific family of planar graphs for which Metric

DIMENSION is essentially a local problem. The algorithm on outerplanar graphs uses a tree structure to traverse the graph, together with several data structures that track the influence of landmarks on other vertices. As we show later, this is sufficient to keep the nonlocality of the problem in check. We believe that our algorithmic techniques are of independent interest, and could lead to (new) algorithms for a broad class of nonlocal problems.

Overview of the Algorithm. Observe that the standard dynamic programming approach using a tree decomposition fails here as the amount of information one needs to maintain seems to depend exponentially on n, rather than on the width of the decomposition. To overcome this fact we take a different approach.

First, we characterize resolving sets in outerplanar graphs by giving two necessary and sufficient requirements for an arbitrary vertex set to be a resolving set. Then, taking as a base the duals of the biconnected components of the graph G, we define a tree T. Vertices of T correspond to faces and cut vertices of G, and edges to inner edges and bridges of G. Note that each vertex of T is a separator and splits the graph into two parts. The algorithm uses dynamic programming to process T, starting at the leaves and advancing towards the root.

At first sight, this decomposition has the same problem as we had with tree decompositions. Moreover, the size of a face might be arbitrarily big, leading to a decomposition of arbitrary 'width'. To overcome these obstacles, we introduce two data structures, called *boundary conditions* and *configurations*.

Boundary conditions track the effects of landmarks placed in the already processed part of the graph and the possible effects of sets of landmarks to be placed in the unexplored parts of the graphs. The main algorithmic novelty lies in the configurations, which control the process of combining the boundary conditions on edges towards children of the current vertex $v' \in V(T)$ into a boundary condition on the edge towards the parent of v'. The configurations depend on the vertices of G represented by v'. Even though the number of vertices of G represented by v' may be unbounded, we show that the total number of relevant configurations is only polynomial.

The use of configurations presents a stark contrast with the techniques used in bounded treewidth algorithms, where the combination process commonly is a simple static procedure. A similar contrast is apparent in our tree structure: whereas outerplanar graphs have constant treewidth [4], the tree structure used in our approach actually leads to a decomposition that can have arbitrary width. Our methods also provide a simpler algorithm to solve the problem on trees.

Preliminaries. For basic notions and results in graph theory, we refer the reader to any textbook on the topic, e.g. [9]. All graphs are finite, undirected, and unless otherwise stated, connected. We use (u, v) to denote an edge from u to v. By distance $d(u, v)$ we mean the graph distance. The vertex and edge sets of G are denoted by $V(G)$ and $E(G)$, respectively.

Recall that a graph is *outerplanar* if and only if it does not contain a subdivision of K_4 or $K_{2,3}$. A graph G has a *cut vertex* if the removal of that vertex disconnects the graph into two components. A graph is a *biconnected* if it has no cut vertices. If G is a biconnected outerplanar graph, G has a planar embedding

in which the edges on the boundary of the outer face form a Hamiltonian cycle. We call those edges *outer edges* and the other edges are called *inner edges*. Given G and $v \in V(G)$, $\mathcal{N}(v)$ denotes the set of neighbors of v in G. Given a set S, we denote by $\mathcal{P}(S)$ the power set of S. Due to the space restrictions many technical details and proofs are omitted. A full version of the paper is available on arXiv.

2 NP-Hardness on Planar Graphs

We reduce from a variation of the 3-SAT problem. We first require some notation. Let Ψ be a boolean formula on a set V of variables and a set C of clauses. The *clause-variable graph* of Ψ is defined as $G_\Psi = (V \cup C, E)$, where $E = \{(v, c) \mid v \in V, c \in C, v \in c\}$. The notation $v \in c$ means that variable v (or its negation) occurs in clause C. Observe that G_Ψ is always bipartite.

It is well known that 3-SAT is NP-complete [14], and that it remains NP-complete even with additional restrictions, such as requiring G_Ψ to be planar [7, p. 877]. As a starting point for our work, we consider the following restrictions. Let 1-Negative Planar 3-SAT be the problem of deciding the satisfiability of a boolean formula Ψ with the following properties: (1) every variable occurs exactly once negatively and once or twice positively, (2) every clause contains two or three distinct variables, (3) every clause with three distinct variables contains at least one negative literal, and (4) G_Ψ is planar.

Lemma 1 1-Negative Planar 3-SAT *is NP-complete.*

To prove that Planar Metric Dimension is NP-hard, we give a reduction from 1-Negative Planar 3-SAT. The idea behind the graph constructed in this reduction is the following. Given an instance Ψ of 1-Negative Planar 3-SAT, we first find a planar embedding of its clause-variable graph G_Ψ. We then replace each variable vertex of G_Ψ by a *variable gadget*, and each clause vertex of G_Ψ by a *clause gadget*. By identifying vertices of variable gadgets and vertices of clause gadgets in an appropriate way, we obtain a planar graph H_Ψ that will be our instance of Planar Metric Dimension.

Let n be the number of variables. Each variable gadget is constructed such that it must contain 4 landmarks: 3 at known, specific locations, but for the fourth we have three different choices. They correspond to the variable being true, false, or undefined. These $4n$ landmarks are a resolving set if and only if they resolve all pairs of vertices in clause gadgets, which happens only if they correspond to a satisfying truth assignment of the SAT-instance. Then we get the following theorem (technical details are deferred to the full version).

Theorem 2 Planar Metric Dimension *is NP-complete, even on bounded-degree graphs.*

3 Algorithm for Outerplanar Graphs

In this section, we prove that Metric Dimension can be solved in polynomial time on outerplanar graphs. So let G be an outerplanar graph, given together

with an outerplanar embedding. Note that a metric base of a disconnected graph is the union of metric bases of its components[3]. So we can safely assume that G is a connected graph. We can also assume that it has at least three vertices.

As a technical trick we sometimes treat the midpoint of an inner edge $e = (v_1, v_2) \in E(G)$ as an actual vertex. The distances from this *midpoint vertex* v_e are such that $d(v_e, v_1) = d(v_e, v_2) = \frac{1}{2}$ and $d(v_e, x) = \min(d(v_e, v_1) + d(v_1, x), d(v_e, v_2) + d(v_2, x))$.

3.1 Characterization of Resolving Sets in Outerplanar Graphs

We first give a characterization of the effects of resolving sets in vertices and faces of outerplanar graphs. To this end, we introduce some notation. A *bifurcation point* associated with z, x, y is a vertex v farthest from z such that v is on shortest path from z to both x and y. More formally, v is a bifurcation point if it is on shortest paths $z \rightsquigarrow x$, $z \rightsquigarrow y$, and if any shortest paths $v \rightsquigarrow x$, $v \rightsquigarrow y$ intersect only in v. Notice that in an outerplanar graph the bifurcation point for each triple of vertices is unique.

We define the function $g : V(G) \times \mathcal{P}(V(G)) \to \mathcal{P}(V(G))$ as

$$g(v, L) = \{w \in \mathcal{N}(v) : d(z, w) = d(z, v) + 1 \text{ for all } z \in L\}.$$

In other words, a neighbor w of v is in $g(v, L)$ if for every $z \in L$, v is on some shortest path $z \rightsquigarrow w$ (but not necessarily on every shortest path.)

Any pair $x, y \in g(v, L)$ is left unresolved by landmarks in L. So any resolving set L satisfies the following:

Requirement 1 *Any vertex $v \in V(G)$ must have $|g(v, L)| \leq 1$.*

If G is a tree, then the requirement is sufficient. The following lemma gives an alternative and simpler correctness proof for the algorithm by Khuller et al. [19].

Lemma 3 *If G is a tree with at least three vertices, then a set $L \subseteq V(G)$ is a resolving set if and only if it satisfies Requirement 1.*

Proof. We have already seen that any resolving set L satisfies Requirement 1. Now assume that Requirement 1 is satisfied. We pick any two vertices $x, y \in V(G)$ and show that they are resolved.

Since G has at least 3 vertices, there is at least one vertex $v \in V(G)$ with degree at least 2. Since $|g(v, L)| \leq 1 < |\mathcal{N}(v)|$, L is not empty.

Choose any landmark $z \in L$. If z resolves x, y, we are done. Otherwise, let v be the bifurcation point associated with z, x, y, and let v_1, v_2 be the successors of v on the shortest paths $v \rightsquigarrow x, v \rightsquigarrow y$. Since $d(z, x) = d(z, y)$, we have $d(v, x) = d(v, y)$. By assumption, $g(v, L)$ can not contain both v_1 and v_2. Without loss of generality $v_1 \notin g(v, L)$. Then there is a landmark z_2 with $d(z_2, v_1) < d(z_2, v)$, and furthermore $d(z_2, x) < d(z_2, y)$. □

[3] With one exception: isolated vertices. An edgeless graph of n vertices has metric dimension $n - 1$.

As stated earlier, the major difficulty of the metric dimension problem is that it is non-local. This is why Lemma 3 is useful. Although stopping short of giving an actual local characterization of resolving sets, it does make the effects of a resolving set local enough to devise a polynomial-time algorithm for trees.

Our algorithm relies on a generalization of Lemma 3 to outerplanar graphs. In this case Requirement 1 no longer implies that L is a resolving set. For example, if G is an even cycle and L contains two antipodal vertices, then Requirement 1 is satisfied, but L is not a resolving set. Therefore we need another requirement that provides an additional condition for the effects of a resolving set on the faces of outerplanar graphs. First, we need some auxiliary definitions and lemmas.

Definition 4 *Let $z \in V(G)$ be a landmark, and let C be either a single edge or a cycle—in particular, it may be a face. The* representative *of z on C is the element of $V(C)$ closest to z, if it is unique. Otherwise outerplanarity implies that there are two closest vertices, which are adjacent. In this case the representative is the midpoint of those two vertices.*

We can make the following observations. Let C, C' be cycles such that $V(C') \subseteq V(C)$ and C' is a face. If two vertices have the same representative on C, then they must have the same representative on C' as well.

Lemma 5 *Let G be a graph, and let $x, y, z, z_2 \in V(G)$. If neither z nor z_2 resolves the pair x, y and there exist intersecting shortest paths $z \rightsquigarrow x$, $z_2 \rightsquigarrow y$, then a bifurcation point of z, x, y is also a bifurcation point of z_2, x, y.*

We are now ready to present the other necessary requirement for a set of landmarks L to be a resolving set.

Requirement 2 *Let v' be a face of an outerplanar graph G in which L has exactly two representatives $\hat{z}_1, \hat{z}_2$ on v'. Let z_1^f and z_1^l be the landmarks with representative $\hat{z}_1$ which occur first and last on the walk along the outer face starting at $\hat{z}_2$, and define z_2^f, z_2^l analogously. Assume that neither z_1^f nor z_2^f resolves a pair x, y, and that shortest paths $z_1^f \rightsquigarrow x$, $z_2^f \rightsquigarrow y$ do not intersect. Let v be the bifurcation point of z_1^f, x, y, let u be the bifurcation point of x, z_1^f, z_2^f, let s be the bifurcation point of y, z_1^f, z_2^f and let t be the bifurcation point of z_2^f, x, y. By assumption, $v \neq t$. Therefore the shortest paths $s \rightsquigarrow t$, $t \rightsquigarrow u$, $u \rightsquigarrow v$, $v \rightsquigarrow s$ form a cycle C. If v' is inside the cycle C, then one of z_1^l, z_2^l must resolve the pair x, y.*

Note that the representatives of z_1^f and z_2^f on C are v and t, respectively.

The assumption that G is outerplanar is essential for Requirement 2 and Lemma 6 below. In particular, the definition of $z_1^f, z_2^l, z_2^f, z_2^l$, as well as the use of representatives in the proof of Lemma 6, relies on outerplanarity.

Lemma 6 *If G is an outerplanar graph, then any resolving set $L \subseteq V(G)$ satisfies Requirement 2.*

Now we are ready to generalize Lemma 3 to outerplanar graphs. This is a crucial result, since it characterizes resolving sets in a manner that allows dynamic programming.

Theorem 7 *If G is an outerplanar graph, then a set $L \subseteq V(G)$ is a resolving set if and only if it satisfies Requirements 1 and 2.*

Proof. We have already seen that any resolving set satisfies the requirements. So assume that $L \subseteq V(G)$ satisfies the Requirements, and choose any $x, y \in V(G)$. We show that there exists a $z \in L$ that resolves the pair x, y.

As in Lemma 3, L is non-empty. Choose $z \in L$ arbitrarily. If z resolves x and y, we are done; so assume that it does not. As in Lemma 3, let v be the bifurcation point of z, x, y, and let v_1, v_2 be successors of v on some shortest paths $v \rightsquigarrow x, v \rightsquigarrow y$ respectively. By Requirement 1, there is a $z_2 \in L$ such that, without loss of generality, $d(z_2, v_1) \leq d(z_2, v)$. If z_2 resolves x and y, we are done. So assume otherwise.

If there exist intersecting shortest paths $z \rightsquigarrow x$, $z_2 \rightsquigarrow y$, then by Lemma 5, v is on a shortest path $z_2 \rightsquigarrow x$. Then v_1 is also on such a path, and $d(z_2, v_1) = d(z_2, v)+1$, a contradiction. Therefore no such pair of intersecting shortest paths exists. Define v, t, C as in Requirement 2, and let v' be a face inside C. If there exists a $z_3 \in L$ whose representative on v' is distinct from the representatives of z_1 and z_2, then its representative on C is neither v nor t. Hence z_3 resolves x, y. If such a landmark z_3 does not exist, then Requirement 2 implies the claim. □

3.2 Data Structures

We now introduce the complex data structures that we need in our algorithms. The *weak dual* of an outerplanar graph G is a graph which has the faces of G, except the unbounded outer face, as vertices. Two faces are adjacent if they share an edge. The weak dual of a biconnected outerplanar graph is a tree.

The *generalized dual tree* $T = (V', E')$ of an outerplanar graph G is defined as follows. T contains the weak dual of G, and also contains the subgraph of G induced by all cut vertices and vertices of degree one. For any cut vertex contained in a biconnected component, there is an edge from the vertex to an arbitrary incident face of the component. Observe that the resulting graph is a tree. According to this definition, a cut vertex is a vertex of both G and T. Let any vertex of T be the root, denoted by v'_r.

We now associate a subset of $V(G)$ with any vertex or edge of T. If $v' \in V(T)$ is a face, the set $s(v')$ consists of the vertices on the face. If v' is a cut vertex or a leaf, $s(v')$ consists of that vertex. If both endpoints of $e' \in E(T)$ are vertices of G, then $s(e')$ consists of those vertices. Otherwise, let either endpoint of e' is a face. Let $e' = (v'_1, v'_2)$, and define $s(e') = s(v'_1) \cap s(v'_2)$.

Removing an edge (v', w') divides T into two components, $T_{v'}$ and $T_{w'}$, where $T_{v'}$ is the one containing v'. Define $B(v', w')$ as the subgraph of G corresponding to $T_{v'}$. Formally, it is the subgraph of G induced by $\bigcup_{u' \in V(T_{v'})} s(u')$. Thus G is divided into two subgraphs $B(v', w')$, $B(w', v')$. If v', w' are adjacent faces, the

subgraphs share two vertices. If v' is a face and w' a cut vertex (or the other way around), then the subgraphs share one vertex. Otherwise they do not intersect. Define $B^-(v', w')$ as the subgraph of G induced by $V(G) \setminus V(B(w', v'))$. Then we can divide G into two nonintersecting subgraphs, $B^-(v', w')$ and $B(w', v')$.

The following lemma is immediate from the definitions.

Lemma 8 *Given neighbors $v', w' \in V(T)$, $B(v', w')$ and $B(w', v')$ are connected subgraphs of G, and any path from $B(v', w')$ to $B(w', v')$ intersects $s((v', w'))$.*

Given a landmark z, let $S(z, W) = \{\{x, y\} \in W \times W : d(z, y) \neq d(z, x)\}$ be the set of pairs in $W \subseteq V(G)$ resolved by z.

Lemma 9 *Let $e' = (v', w') \in E(T)$. Assume $z \in V(B(v', w'))$ and denote $W = V(B(w', v'))$. Then $S(z, W) = S(\hat{z}, W)$, where $\hat{z}$ is the representative of z on e'.*

Let v' be a dual vertex and p' its parent. We use *boundary conditions* to describe the relation of landmarks in $B(v', p')$ to $B(p', v')$, and vice versa. Boundary conditions can be seen as an abstract data structure which depends on v', p', L and satisfies the following:

1. It consists of two components, one of which depends on landmarks in $B(p', v')$, and the other on landmarks in $B^-(v', p')$. The components are called *upper* and *lower* boundary conditions, respectively.
2. It determines which pairs in $B(p', v')$ are resolved by landmarks in $B^-(v', p')$, and vice versa.
3. If $B(v', p')$ contains landmarks with a representative v on v', then the boundary condition specifies which such landmarks occur first and last on a walk along the outer boundary of $B(v', p')$ starting at v.
4. For any $v \in s((p', v'))$, it specifies whether the set $g(v, L) \cap V(B^-(v', p'))$ is empty or not.
5. The number of possible boundary conditions is polynomial.

The first and the second properties are necessary to be able to run a dynamic programming algorithm along T. The third and fourth properties are needed when verifying Requirements 2 and 1, respectively. The last property is needed to ensure that the algorithm runs in polynomial time.

While a boundary condition describes how the landmarks are placed in relation to a dual edge e', a *configuration* describes their relation to a dual vertex v'. There are three quite different cases, depending on whether v' is a cut vertex of G, a face with two representatives, or a face with at least three representatives.

A configuration L associated with v' is designed to satisfy following:

1. For any $w' \in \mathcal{N}(v')$, it determines which pairs of vertices in $B(w', v')$ are resolved by landmarks in $B(v', w')$.
2. It contains enough information to verify that v' satisfies Requirement 2.
3. The total number of configurations is polynomial.

Essentially, a configuration determines which vertices of $s(v')$ are landmarks, and which boundary conditions are allowed for edges (v', w').

Even if both boundary conditions for edges (v', w_1'), (v', w_2') are allowed by a specific configuration, they may contradict each other. This happens when there is $v \in s((v', w_1')) \cap s((v', w_2'))$, and the boundary conditions disagree about the value of $g(v, L)$. Hence, the algorithm will only process boundary conditions that agree with each other and with the configuration.

As a tool for avoiding such disagreements, we define a coarser variant of the function $g(\cdot, \cdot)$. The function $h : V(G) \times V(T) \times \mathcal{P}(V(T)) \to V(T) \cup \{\emptyset\}$ describes which part of the generalized dual contains $g(v, L)$. Let $v \in s(v')$. Then

$$h(v, v', L) = \begin{cases} v' & \text{if } g(v, L) \cap s(v') \neq \emptyset, \\ w' & \text{if } w' \text{ is neighbor of } v' \text{ and } g(v, L) \cap V(B^-(w', v')) \neq \emptyset, \\ \emptyset & \text{if } g(v, L) = \emptyset. \end{cases}$$

Notice that as long as Requirement 1 holds, the function is well defined.

Lemma 10 *The configuration associated with v' and L determines for any $v \in s(v')$ whether the equation $h(v, v', L) = v'$ holds.*

3.3 Algorithm

The algorithm works in a dynamic programming manner, by finding optimal resolving sets of subgraphs $B(v', p')$ with given boundary conditions. Formally, assume that we are given a vertex $v' \in V(T)$ and boundary condition $\mathbf{r}$ on the edge $e' = (v', p')$, where p' is the parent of v'. Let $m(v', \mathbf{t}) \subseteq V(B^-(v', p'))$ be a set $L \cap V(B^-(v', p'))$, where L is a minimal resolving set with boundary condition $\mathbf{t}$, if such a set exists. Otherwise $m(v', \mathbf{t}) = \text{NIL}$. For notational convenience, we let $|\text{NIL}| = \infty$ and $\text{NIL} \cup A = \text{NIL}$ for any A.

The values of $m(v', \mathbf{t})$ are computed in a recursive manner: the computation of $m(v', \mathbf{t})$ uses the values of $m(w', \mathbf{r})$, where w' is a child of v'. The basic idea is to iterate over all configurations on v'. For every configuration, we then find an optimal function h and an optimal set of landmarks.

First, we introduce several subroutines. Algorithm 1 (Intermediate-sol) returns, given a configuration (v', R), a boundary condition $\mathbf{t}$, and a function h, an optimal set of landmarks.

Given a configuration (v', R) and a boundary condition $\mathbf{t}$, the next subroutine finds an optimal function h. By Lemma 10, the configuration determines whether $h(v, v', L) = v'$ holds or not. Also $\mathbf{t}$ restricts some values of h. Otherwise, the value of $h(v, v', L)$ may be $\emptyset$ or w', where w' is a suitable neighbor of v', and the task is to determine which one of these is optimal. It turns out that the optimum can be found by a greedy algorithm (Algorithm 2).

Lemma 11 *Algorithm 2 runs in polynomial time and returns a smallest resolving set of $B(v', p')$ that agrees with the parameters.*

We just emphasize that the greedy algorithm relies on the following observation about cardinalities of optimal solutions L_1, L_2 with $h(v, v', L_1) = \emptyset$,

Algorithm 1 Intermediate-sol

Input: Configuration (v', R), boundary condition $\mathbf{t}$, function h
  if the parameters are inconsistent or (v', R) violates Requirement 2 **then**
    return NIL
  end if
  Initialize W to the set of landmarks on $s(v')$ as described by (v', R)
  for all w' that are children of v' **do**
    $\mathbf{r} \leftarrow \arg\min_{\mathbf{r}} m[w', \mathbf{r}]$ such that $\mathbf{r}$ agrees with $C, h, \mathbf{t}$
    $W \leftarrow W \cup m[w', \mathbf{r}]$
  end for
  return $W \cap V(B^-(v', p'))$

$h(v, v', L_2) = w'$. Let $g(v, L_2) = \{w\}$ and note that $M_1 = L_2 \cup \{w\}$ is a solution with $h(v, v', M_1) = \emptyset$. Therefore $|L_1| \leq |M_1| = |L_2| + 1$.

Finally, we are ready to present the main algorithm (Algorithm 3) and its correctness proof.

Theorem 12 *Algorithm 3 correctly computes the values of $m[v', \mathbf{r}]$, and returns a metric base in polynomial time.*

Proof. The algorithm runs in polynomial time, because there is a polynomial number of dual vertices, boundary conditions, and configurations.

We prove by induction that $m[v', \mathbf{r}]$ is computed correctly. Assume that the values of $m[w', \mathbf{t}]$ have been computed correctly for any child w' of v'. Then, because Opt (Algorithm 2) works correctly, the value of $m[v', \mathbf{r}]$ will be computed correctly. Similarly, the return value will be computed correctly, since Opt works correctly and uses correct values of $m[w', \mathbf{r}]$. □

4 Conclusions and Open Problems

We have showed that Metric Dimension is NP-hard for planar graphs, even when the graph has bounded degree (an open problem from 1976). We also gave a polynomial-time algorithm to solve the problem on outerplanar graphs. Our algorithm is based on an innovative use of dynamic programming which allows us to deal with the non-bidimensional, global problem of Metric Dimension.

We pose some open problems about Metric Dimension. First, it would be nice to extend our results to k-outerplanar graphs. The main obstacle to extending the result is that the separators to be associated with nodes of the computation tree should include faces and edges between consecutive levels. For such separators we lose the relevant property that shortest paths between nodes in different parts cross the separator only once.

Even if the problem turns out to be solvable on k-outerplanar graphs by a polynomial-time algorithm, it is not clear that such an algorithm could be used to derive a polynomial-time approximation scheme for Planar Metric Dimension. The quest for such an approximation scheme or even for a constant approximation algorithm is an interesting challenge in its own right.

Algorithm 2 Opt

Input: Configuration (v', R), boundary condition $\mathbf{t}$
  $Q \leftarrow \emptyset$ {Q is the set of vertices for which $h[v]$ is already fixed}
  for all $v \in s(v')$ **do**
    if R or $\mathbf{r}$ determine $h(v, v', L)$ **then**
      $h[v] \leftarrow$the appropriate value
      $Q \leftarrow Q \cup \{v\}$
    else
      $h[v] \leftarrow \emptyset$
    end if
  end for
  for all w' that are children of v', in clockwise order, starting on the successor of p' **do**
    $P \leftarrow s((v', w')) \setminus Q$
    Find k that minimizes |Intermediate-sol$(v', R, \mathbf{r}, k)$| such that
    $h[v]$ and $k[v]$ differ only for $v \in P$. If possible, choose a solution with $k[v_l] = \emptyset$ for the last (in clockwise order) element $v_l \in P$
    $h \leftarrow k$
    $Q \leftarrow Q \cup \{v : h[v] \neq \emptyset\}$
  end for
  return Intermediate-sol$(v', R, \mathbf{r}, h)$

Algorithm 3 Metric-D

Input: Graph G, its generalized dual tree T
  for all $v' \in V(T) \setminus \{v'_r\}$, boundary condition $\mathbf{r}$ **do** {recall that v'_r is the root}
    $m[v', \mathbf{r}] \leftarrow$ NIL
  end for
  for all $v' \in V(T) \setminus \{v'_r\}$, in postorder **do**
    $p' \leftarrow$ the parent of v'
    for all configuration (v', R) **do**
      for all boundary condition condition $\mathbf{r}$ on (v', p') **do**
        if $|\text{Opt}(v', R, \mathbf{r})| \leq |m[v', \mathbf{r}]|$ **then**
          $m[v', \mathbf{r}] \leftarrow \text{Opt}(v', R, \mathbf{r})$
        end if
      end for
    end for
  end for
  $W \leftarrow V(G)$
  for all configuration (v'_r, R) **do**
    if $|\text{Opt}(v'_r, R, \text{NIL})| \leq |W|$ **then**
      $W \leftarrow \text{Opt}(v'_r, R, \text{NIL})$
    end if
  end for
  return W

Another interesting line of research is the *parameterized complexity* of Metric Dimension. Daniel Lokshtanov [20] posed this problem at a recent Dagstuhl seminar on parametrized complexity. Moreover, he conjectured that the problem could be $W[1]$-complete. We hope that the insights of this paper can help to obtain results in this direction.

Acknowledgment: The authors thank David Johnson for sending a copy of the NP-completeness proof.

References

1. R.F. Bailey, P.J. Cameron Base size, metric dimension and other invariants of groups and graphs. *Bulletin London Math. Society* 43 (2): 209-242, 2011.
2. Z. Beerliova, T. Eberhard, T. Erlebach, A. Hall, M. Hoffmann, M. Mihalak, L. Ram, Network Discovery and Verification. *IEEE J. Selected Areas in Communication* 24, 2168–2181, 2006.
3. B. Baker, Approximation algorithms for NP-complete problems on planar graphs. *JACM* 41, 153–180, 1994.
4. H. L. Bodlaender, Classes of Graphs with Bounded Treewidth. *Bulletin of the EATCS* 36, 116–125, 1988.
5. B. Courcelle, Graph rewriting: An algebraic and logic approach. *Handbook of Theoretical Computer Science*, vol. B, 194–242, Elsevier Science, 1990.
6. G. Chartrand, L. Eroh, M.A. Johnson, O.R. Oellemann, Resolvability in graphs and the metric dimension of a graph. *Discrete Applied Math.* 105, 99–113, 2000.
7. E. Dahlhaus, D.S Johnson, C.H. Papadimitriou, P.D. Seymour, M. Yannakakis, The Complexity of Multiterminal Cuts. *SIAM J. Computing* 23, 864–894, 1994.
8. E.D. Demaine, M.T. Hajiaghayi, Bidimensionality: new connections between FPT algorithms and PTASs. *SODA 2005*, 590-601, 2005.
9. R. Diestel, *Graph Theory.* Springer-Verlag, 2000.
10. L. Epstein, A. Levin, G.J. Woeginger, The (weighted) metric dimension of graphs: hard and easy cases. *Proc. WG 2012*, to appear.
11. L.H Erickson, J. Carraher, I. Choi, M. Delcourt, D.B. West, Locating a robber on a graph via distance queries, preprint.
12. D. Eppstein, Diameter and treewidth in minor-closed graph families. *Algorithmica* 27, 275–191, 2000.
13. F.V. Fomin, D. Lokshtanov, V. Raman, S. Saurabh, Bidimensionality and EPTAS. *SODA 2011*, 748-759, 2011.
14. M.R. Garey, D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness.* Freeman, 1979.
15. F. Harary, R.A. Melter, The metric dimension of a graph. *Ars Combinatoria* 2, 191–195, 1976.
16. M. Hauptmann, R. Schmied, C. Viehmann, On approximation complexity of metric dimension problem. *J. Discrete Algorithms*, in press, 2011.
17. J. Hopcroft, R.E. Tarjan, Efficient planarity testing. *JACM* 21, 549–568, 1974.
18. D.S. Johnson, personal communication.
19. S. Khuller, B. Raghavachari, A. Rosenfeld, Landmarks in Graphs. *Discrete Applied Math.* 70, 217–229, 1996.
20. D. Lokshtanov, Metric Dimension. In: Open Problems from Dagstuhl Seminar 09511, E. D. Demaine, M. T. Hajiaghayi, D. Marx, editors, 2009. Available at http://erikdemaine.org/papers/DagstuhlFPT2009Open/paper.pdf
21. P. Slater, Leaves of trees. *Congressus Numerantium* 14, 549–559, 1975.