Sequential Metric Dimension

Julien Bensmail, Dorian Mazauric, Fionn Mc Inerney, Nicolas Nisse, Stéphane Pérennes

Sequential Metric Dimension*

Julien Bensmail[1], Dorian Mazauric[2], Fionn Mc Inerney[1], Nicolas Nisse[1], and Stéphane Pérennes[1]

[1]Université Côte d'Azur, Inria, CNRS, I3S, France
[2]Université Côte d'Azur, Inria, France

Abstract

In the localization game, introduced by Seager in 2013, an invisible and immobile target is hidden at some vertex of a graph G. At every step, one vertex v of G can be probed which results in the knowledge of the distance between v and the secret location of the target. The objective of the game is to minimize the number of steps needed to locate the target whatever be its location.

We address the generalization of this game where $k \geq 1$ vertices can be probed at every step. Our game also generalizes the notion of the *metric dimension* of a graph. Precisely, given a graph G and two integers $k, \ell \geq 1$, the Localization problem asks whether there exists a strategy to locate a target hidden in G in at most ℓ steps and probing at most k vertices per step. We first show that, in general, this problem is NP-complete for every fixed $k \geq 1$ (resp., $\ell \geq 1$). We then focus on the class of trees. On the negative side, we prove that the Localization problem is NP-complete in trees when k and ℓ are part of the input. On the positive side, we design a $(+1)$-approximation algorithm for the problem in n-node trees, *i.e.*, an algorithm that computes in time $O(n \log n)$ (independent of k) a strategy to locate the target in at most one more step than an optimal strategy. This algorithm can be used to solve the Localization problem in trees in polynomial time if k is fixed.

We also consider some of these questions in the context where, upon probing the vertices, the relative distances to the target are retrieved. This variant of the problem generalizes the notion of the *centroidal dimension* of a graph.

Keywords: Games in graphs, Metric dimension, Complexity.

1 Introduction

Unless stated otherwise, every graph considered in this paper is assumed to be connected, undirected, and simple. *Localization* (or *Identification*) *problems* consist of distinguishing the vertices of a graph $G = (V, E)$ using a smallest subset $R \subseteq V$ of its vertices. Many variants have been studied depending on how R is required to make the vertices distinguishable. For instance, *identifying codes* [16], *adaptive identifying codes* [2], and *locating dominating sets* [21] ask for the vertices to be distinguished by their neighbourhood in R. Another well studied example is the one of *resolving sets* [13, 20], where one aims at distinguishing the vertices of a graph by their distances to such a set. Given a graph G, the main problem is to compute a resolving set with minimum size, this minimum being

*This work has been partially supported by ANR program "Investments for the Future" under reference ANR-11-LABX-0031-01, the Inria Associated Team AlDyNet. An extended abstract of parts of this paper has been presented in [3].

called the *metric dimension* of G [13, 20]. The corresponding decision problem (first shown to be NP-complete in [12]) is NP-complete in planar graphs [8] and in graphs of diameter 2 [11], and W[2]-hard (parameterized by the solution's size) [14]. On the positive side, the problem is FPT in the class of graphs with bounded treelength [1]. Bounds on the metric dimension have also been determined for various graph classes [10].

In this paper, we address a *sequential variant* of this problem, which we deal with through the following terminology. Let us consider a graph $G = (V, E)$ where an unknown vertex $t \in V$ hosts a hidden (invisible) and immobile target. *Probing* one vertex $v \in V$ results in the knowledge of the distance between t and v, denoted by $d_G(v, t)$, which is the length of a shortest path from t to v. Probing a set $R \subseteq V$ of vertices results in the *distance vector* $(d_G(v, t))_{v \in R}$ and R is *resolving* if no two vertices of G get the same distance vector (by R). The *metric dimension* of G, denoted by $MD(G)$, is then the minimum number of vertices that must be probed simultaneously to immediately (in one step) determine the location t of the target (wherever it is). For instance, in the case of a path, probing one of its ends is sufficient to locate the target, *i.e.*, $MD(P) = 1$ for every path P. Another example is the case of a star (tree with a universal node) with n leaves, denoted by S_n, for which it is necessary and sufficient to probe every leaf but one, *i.e.*, $MD(S_n) = n - 1$.

If less than $MD(G)$ vertices can be probed at once, then it is impossible to locate a target in one step, in which case it is natural to allow more than one probing step. Obviously, if at most $1 \leq k < MD(G)$ vertices can be probed at once, then it is always feasible to locate an immobile target in $\lceil MD(G)/k \rceil$ steps, simply by considering a smallest resolving set R of G, and probing all vertices of R through successive steps (probing at most k vertices each step). However, there are graphs for which the target can be located much faster (see Section 2 or Lemma 4.2). In [18], Seager initiated the study of the following sequential locating game: an invisible and immobile target is hidden at some vertex t, and, at every step, one vertex can be probed to retrieve its distance to t, and the objective is to locate t using the minimum number of steps. Seager gave bounds and exact values on this minimum number of steps in particular subclasses of trees (*e.g.*, subdivisions of caterpillars) [18] but left the problem open in trees in general. In this paper, we study the generalization of this game where $k \geq 1$ vertices can be probed at every step.

Precisely, let $k \geq 1$ be an integer and let $G = (V, E)$ be a graph hosting an invisible and immobile target hidden at $t \in V$. A *k-strategy* is a sequence of probing steps, where, at each step, at most k vertices are probed, and at the end of which t is uniquely determined. Note that, in a k-strategy, the choice of the vertices to be probed at some step obviously depends on the result of the previous steps. Let $\lambda_k(G)$ denote the minimum integer h such that there exists a k-strategy for locating the target in G in at most h steps, whatever be the location of the target. Given G and $k, \ell \geq 1$, the Localization problem asks whether $\lambda_k(G) \leq \ell$. We also consider the dual parameter $\kappa_\ell(G)$ defined as the minimum integer h such that there exists an h-strategy for locating the target in G in at most ℓ steps. Note that, for every graph G, the parameter $\kappa_1(G)$ is exactly the metric dimension $MD(G)$ of G, and $\lambda_k(G) \leq \ell$ if and only if $\kappa_\ell(G) \leq k$. We are interested in the complexity of the Localization Problem in general graphs and particularly in trees. Note that by the remarks above (Lemma 4.2), the Localization Problem and Metric Dimension Problem (for which $\ell = 1$) behave very differently, so knowing that the Metric Dimension Problem is NP-complete does not imply the same for the Localization Problem.

1.1 Related work

Moving target Sequential games related to resolving sets have first been introduced and studied in the case of a mobile target. That is, at every step, some vertices may be probed and, if the target has not been located yet, it may move to one of its neighbours [17]. Restrictions on this move are sometimes imposed, such as forbidding the target to "backtrack", *i.e.*, move to a neighbour that has just been probed. This condition is crucial, as allowing backtracking may make the localization of the target impossible in some contexts. For instance, it is not possible to locate a moving target that is allowed to backtrack in a triangle when probing one vertex per step.

The question of how many times all the edges of a graph must be subdivided to ensure locating a moving target probing one vertex (resp., k vertices) per step has been addressed in [7] (resp. [15]). A graph is called *locatable* if there exists a 1-strategy for locating a target that is not allowed to backtrack in a finite number of steps. Locatable trees were first studied in [17], where it was proved that all trees are locatable, and a first bound on the number of steps it takes to locate the target was exhibited. This upper bound was then improved in [6]. In [19], the case of trees with a target allowed to backtrack was considered. Let $\zeta(G)$ be the minimum integer k such that there exists a k-strategy for locating a moving target in G. In [5], it was shown that deciding whether $\zeta(G) \leq k$ is NP-hard and that $\zeta(G)$ is not bounded in the class of graphs G with treewidth 2. Moreover, $\zeta(G) \leq 3$ for any outerplanar graph G [4].

Relative distances and centroidal dimension Foucaud *et al.* defined a variant of resolving sets, called *centroidal bases*, where the vertices of a graph must be distinguished by their *relative distances* to the probed vertices [9]. In this setting, given an integer $k \geq 2$, probing a set $B = \{v_1, \ldots, v_k\}$ of vertices results in the *relative-distance vector* $(\delta_{i,j}(t))_{1 \leq i < j \leq k}$ where, for every $1 \leq i < j \leq k$, $\delta_{i,j}(t) = 0$ if $d_G(t, v_i) = d_G(t, v_j)$, $\delta_{i,j}(t) = 1$ if $d_G(t, v_i) > d_G(t, v_j)$, and $\delta_{i,j}(t) = -1$ otherwise. Intuitively speaking, the relative-distance vector of t indicates which vertices of B are the closest to t, which vertices are the second closest, etc., without indicating the exact distances between v and these vertices. The set B is a *centroidal basis* of G if the relative-distance vectors are distinct for every two vertices of G. The *centroidal dimension* of G, denoted by $CD(G)$, is the minimum size of a centroidal basis of G [9]. Note that $CD(G) \geq 2$ unless G has only one vertex, and that $CD(G)$ is well defined since, clearly, V is a centroidal basis of G. The decision problem associated to the centroidal dimension was shown to be NP-complete, and almost tight bounds on the centroidal dimension of paths have been computed (see [9]).

Again, sequential variants of the centroidal basis can naturally be defined, where, at each step, the relative-distance vector of t obtained from the probing of the k vertices at that step is returned. The variant where the target is allowed to move was considered in [4]. In this work, we also initiate the study of the variant where the target is immobile, which, to the best of our knowledge, has not been considered yet. Note that, since the target is immobile and at least $k \geq 2$ vertices are probed at each step, at least one vertex is removed from the possible locations of the target at each step (at least one of the $k \geq 2$ vertices is farther from the target than the others, in which case it is removed, or all of the $k \geq 2$ vertices are at the same distance from the target, in which case they are all removed). Let $k \geq 2$ be an integer and G be a graph. Let $\lambda_k^{rel}(G)$ denote the minimum integer h such that there exists a k-strategy for locating, through the relative-distance vectors, a hidden immobile target in G in at most h steps, whatever be its location. Given G, k, ℓ, the Relative-Localization problem asks whether $\lambda_k^{rel}(G) \leq \ell$. The dual parameter $\kappa_\ell^{rel}(G)$ is defined as the minimum integer h such that there exists an h-strategy for locating,

through the relative-distance vectors, the target in G in at most ℓ steps. Note that, for every graph G, the parameter $\kappa_1^{rel}(G)$ is exactly the centroidal dimension $CD(G)$ of G, and $\lambda_k^{rel}(G) \leq \ell$ if and only if $\kappa_\ell^{rel}(G) \leq k$.

1.2 Our results

This work is dedicated to the computational complexity of the Localization problem, where one aims at locating an invisible and immobile target in a graph through successive probing steps where the distance vectors are retrieved. So that the readers get a first intuition for this problem, we start, in Section 2, by providing first, some observations. In Section 3, we then show that the Localization problem is polynomial-time solvable when both k and ℓ are fixed parameters but that, in general, the Localization problem is NP-complete when only one of k and ℓ is a fixed parameter. Precisely:

- Let $k \geq 1$ and $\ell \geq 1$ be two fixed integers. Given a graph G as an input, the problem of deciding whether $\lambda_k(G) \leq \ell$ is polynomial-time solvable (in time $n^{O(k\ell)}$) (Theorem 3.2).
- Let $k \geq 1$ be a fixed integer. Given a graph G with a universal vertex and an integer $\ell \geq 1$ as inputs, the problem of deciding whether $\lambda_k(G) \leq \ell$ is NP-complete (Theorem 3.3).
- Let $\ell \geq 1$ be a fixed integer. Given a graph G with a universal vertex and an integer $k \geq 1$ as inputs, the problem of deciding whether $\kappa_\ell(G) \leq k$ is NP-complete (Theorem 3.7).

The proof of Theorem 3.2 also yields that the Relative-Localization problem is polynomial-time solvable when $k \geq 2$ and $\ell \geq 1$ are fixed integers. Through modifications, our proofs also yield that the Relative-Localization problem is NP-complete for any fixed $k \geq 2$ (Theorem 3.6) or any fixed $\ell \geq 1$ (Theorem 3.9).

In Section 4, we then focus on the Localization problem in the class of trees. Although we prove that the problem remains NP-complete in the class of trees, surprisingly we show that this hardness only comes from the first probing step. More precisely, we show that, in a tree, Localization becomes polynomial-time solvable after the first step. As a consequence, we design a polynomial-time (+1)-approximation algorithm for the problem. To summarize:

- deciding whether $\lambda_k(T) \leq \ell$ is NP-complete for a tree T when both k and ℓ are part of the input (Theorem 4.3);
- there exists an algorithm that computes, in time $O(n \log n)$ (independent of k), a k-strategy for locating a target in at most $\lambda_k(T) + 1$ steps in any (possibly edge-weighted) n-node tree T (Theorem 4.12);
- deciding whether $\lambda_k(T) \leq \ell$ for any (possibly edge-weighted) n-node tree T can be solved in time $O(n^{k+2} \log n)$ (independent of ℓ) (Theorem 4.13).

2 Preliminaries

For any two vertices $u, v \in V$, we denote by $N_G(v)$ (or simply $N(v)$ when no ambiguity is possible) the set of neighbours of v. Assuming a vertex of G hosts an invisible and

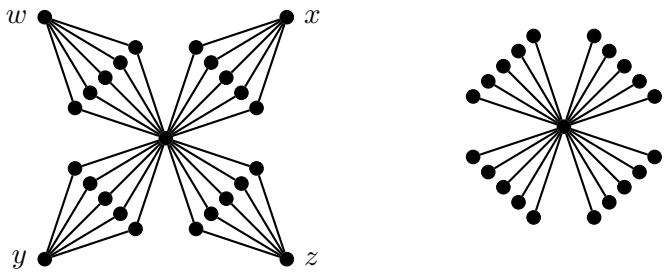

Figure 1: A graph G (left) and an isometric subgraph H of G (right).

immobile target, recall that a *k-strategy* Φ is a sequence of steps where at most k vertices are probed per step, resulting in the exact localization of the target. As we mainly focus on the Localization problem in this work, unless stated otherwise, such a strategy will always deal with the exact distances between the target and the probed vertices. After the s^{th} step of Φ, we denote by $H_s \subseteq V$ the set of vertices that remain as possible locations for the target, *i.e.*, that have not been eliminated at step s. Unless stated otherwise, we thus have $H_0 = V$.

Let us precisely describe the (already mentioned) case of stars because the simple arguments occurring in this case will be used as basic tools for several of the proofs in this paper. Given a star S_n with n leaves, $\lambda_k(S_n) = \lceil \frac{n-1}{k} \rceil$ and any optimal strategy to locate the target consists of probing every leaf but one. Indeed, if the target is at distance 1 of a leaf, then it is located at the center of the star. Otherwise, if the target is at distance 2 from each of the probed leaves, it must be located in the single unprobed leaf. On the other hand, if at least two leaves have not been probed, there is no way to decide in which unprobed leaf the target is located.

The Relative-Localization problem slightly differs since, in this variant, all leaves must be probed. Indeed, after having probed all leaves but one, a last probe is necessary to decide whether the target occupies the last (unprobed) leaf or the center of the star.

To conclude this section, let us observe the following properties. They will not be used further in this paper, however, we believe that they are interesting by themselves and give some hints on the difficulty of designing a strategy for locating a target.

First, let us notice that the metric dimension is not closed under isometric subgraphs. That is, there exists a graph G having an isometric subgraph H such that $MD(H) > MD(G)$. Recall that H is an isometric subgraph of G if, for all $u, v \in V(H)$, $d_G(u, v) = d_H(u, v)$. That is, the distances in H are the same as in G. Let H be the star S_4 and let G be obtained from H by adding two adjacent vertices u and v each adjacent to a different leaf of H and a vertex w adjacent to one of the two remaining leaves of H. In this case, $MD(H) = 3$ and $MD(G) = 2$ (by probing u and the only leaf in G that is not w). This kind of result is also true for our parameters.

Observation 2.1. *For all integers $k \geq 1$, there exists a graph G having an isometric subgraph H such that $\lambda_k(H) > \lambda_k(G)$.*

Proof. Let $q \geq 2$ and let $H_{k,q}$ be the star $S_{(k+1)q}$ with center c and $(k+1)q$ leaves $v_1, \ldots, v_{(k+1)q}$. Let $G_{k,q}$ be the graph obtained from $H_{k,q}$ by adding q vertices $s_1, \ldots, s_q$ such that s_i is adjacent to $v_{(i-1)(k+1)+1}, \ldots, v_{i(k+1)}$ for every $1 \leq i \leq q$. The graphs $G_{4,4}$ and $H_{4,4}$ are depicted in Fig. 1. Clearly, $H_{k,q}$ is an isometric subgraph of $G_{k,q}$ (*i.e.*, distances of $G_{k,q}$ are preserved in $H_{k,q}$).

By the paragraph above, $\lambda_k(H_{k,q}) = \lceil \frac{q(k+1)-1}{k} \rceil$. On the other hand, $\lambda_k(G_{k,q}) \leq \lceil \frac{q}{k} \rceil + 1$ as proved by the following strategy. Probe sequentially every vertex in $s_1, \ldots, s_q$. If, during the first step, the target is at distance 2 from the probed vertices, then the target is in c. Otherwise, if, at some step $t \leq \lceil \frac{q}{k} \rceil$, the target is at distance 0 from some s_j, the target is at s_j. Finally, if, at step $t \leq \lceil \frac{q}{k} \rceil$, the target is at distance 1 from some s_j, then probe the vertices $v_{(j-1)(k+1)+1}, \ldots, v_{j(k+1)-1}$ to locate the target. $\diamond$

As stated in the introduction, there is a strong connection between the metric dimension and our sequential game. For instance, one k-strategy for locating a target in a graph G consists of considering a minimum resolving set R of G, and probing all vertices of R in $\lceil MD(G)/k \rceil$ steps. In general though, this strategy can be arbitrarily far from being optimal. As an illustration, note that for the graphs $G_{k,q}$ constructed in the proof of Observation 2.1, we have $MD(G_{k,q}) = (k+1)q - 1$ (all vertices that are leaves in $H_{k,q}$ must be probed but one), while $\lambda_k(G_{k,q}) \leq \lceil \frac{q}{k} \rceil + 1$.

3 General complexity of Localization and Relative-Localization

In this section, we prove that the (Relative) Localization problem is polynomial-time solvable when both k and ℓ are fixed but NP-complete when only one of k and ℓ is fixed. The proof when ℓ is fixed is an almost straightforward reduction from the Metric Dimension problem. In the case when k is fixed, the proof is a much more involved reduction from the 3-Dimensional Matching problem. Our proofs, through several modifications, also apply to the Relative-Localization problem. The proof that the (Relative) Localization problem is in NP is given as a separate claim (Claim 3.1) as it is used in all of the NP-completeness proofs.

Claim 3.1. *The (*Relative) Localization Problem is in NP.

Proof of claim. The proof is done for the Localization Problem. The certificate is a k-strategy which can be described by a rooted decision tree T as follows. The nodes of T are labelled by sets of k vertices (the vertices to be probed at a given step) and its edges are labelled by sets of vertices representing the possible locations of the target. Precisely, the root node represents the first k vertices to be probed in G according to the k-strategy. For every node $v \in V(T)$ (but the root), the label $L_e \subseteq V(G)$ of the parent-edge e of v represents the current possible locations of the target and the label $L_v \subseteq V(G)$, $|L_v| \leq k$, is the set of vertices to be probed according to the strategy, given that the target occupies a vertex in L_e. Then, every child w of v corresponds to a possible outcome (after probing the vertices in L_v). That is, L_{vw} is the new set of possible locations after having probed L_v (given that the target was in L_e). Note that, clearly, $L_{vw} \subseteq L_e$. Moreover, we may restrict our attention to *progressive* strategies, *i.e.*, strategies for which, for every non-root vertex v with parent-edge e, and for every child-edge f of v, $L_f \subset L_e$. Indeed, otherwise, the vertices probed in L_v are not relevant and a better choice would be any subset containing at least one vertex of L_e (two vertices of L_e in the case of the Relative Localization Problem, where by definition $k \geq 2$, and this is the only part of the proof that differs between the two problems).

The previous remark shows that we can restrict ourselves to k-strategies represented by rooted trees where all non-leaf nodes have at least two children. Moreover, any such tree representing a winning strategy (a k-strategy that locates the target) has exactly $|V(G)|$ leaves since there is a one-to-one correspondence between a path from the root to a leaf of T with the location of the target in G. Indeed, each probe (corresponding to, say, vertex

v of the decision tree) leads to a partition of current possible locations for the target, and therefore, each position of the target can only lead to a unique child of the vertex v. A trivial induction on $|V(T)|$ allows to show that any rooted tree with n leaves and where all non-leaf nodes have at least two children, has at most $2n$ nodes. Thus, any winning k-strategy may be encoded polynomially and the Localization Problem is in NP. $\diamond$

Theorem 3.2. *Let $k \geq 1$ ($k \geq 2$ for the* Relative Localization *problem) and $\ell \geq 1$ be two fixed integers. The (*Relative*)* Localization *problem is polynomial-time solvable (in time $n^{O(k\ell)}$).*

Proof. Let G be any n-node graph. Let us consider the following tree T that will be used to represent all possible strategies that probe exactly k vertices per step and last at most ℓ steps in G. Roughly, this tree is a generalization of the decision tree presented in the proof of Claim 3.1.

With the exception of the root node, the nodes of T are labelled by sets of k vertices (the vertices to be probed at a given step) and its edges are labelled by sets of vertices representing the possible locations of the target. Precisely, so that T is a rooted tree, the root node r (unlike in Claim 3.1) is "artificial" and $L_r = \emptyset$ (its label is empty) since it is as if no vertices are probed on the first turn. For every node $v \in V(T)$ (but the root), the label $L_e \subseteq V(G)$ of the parent-edge e of v represents the current possible locations of the target and the label $L_v \subseteq V(G)$, $|L_v| \leq k$, is the set of vertices to be probed (according to one of the strategies), given that the target occupies a vertex in L_e. Note that $L_{rc} = V(G)$ for all children c of the root r. Every child w of v corresponds to a possible subsequent probing depending on the outcome after probing the vertices in L_v. Then, L_{vw} is the new set of possible locations after having probed L_v (given that the target was in L_e). Note that, clearly, $L_{vw} \subseteq L_e$. Precisely, given the set L_e of possible current locations of the target, and the set L_v of the vertices that have been just probed, this leads to a partition of L_e depending on L_v for every location $t \in L_e$ of the target.

Any non-leaf vertex $v \in V(T)$ has at most $n\binom{n}{k}$ children. Precisely, there are at most n possible different labels for the children-edges of v since there are n possible locations for the target. For each of these at most n possible different labels, there are $\binom{n}{k}$ possible subsequent probings, and so, for each of the subsets of size k of $V(G)$, there is a child labelled with that subset. Since only strategies of length ℓ are being searched for, and since the root node of the tree represents a void probing step, all of the leaves of T are at distance ℓ from the root.

First, note that $|V(T)|$ is polynomial in n when k and ℓ are fixed. Precisely, since T has at most $(\binom{n}{k}n)^\ell$ leaves (due to the degree of the nodes and the height of T) and all leaves are at distance ℓ from r, $|V(T)|$ is upper bounded by $O(\ell(\binom{n}{k}n)^\ell) = n^{O(k\ell)}$.

Secondly, every strategy (of length ℓ and probing k vertices per turn) is "contained" in T. Indeed, any subtree T' of T built as follows represents a strategy: start with T' reduced to a child c of the root r, then while possible, for any leaf v of T' and for each unique child-edge label of v (each possible outcome for the remaining possible locations of the target after the probing of the set of vertices that is $L(v)$ and based on the remaining possible locations defined by the label of the parent-edge of v), add a single child of v to T' (this is the probing that the strategy performs based on that possible outcome). It is easy to see that, in this way, any strategy, winning (locating the target in at most ℓ turns, wherever it is) or not, can be represented.

By the same reasoning, for every node v at distance $\ell - \ell'$ from r, the subtree of T rooted in v, along with the parent-edge pv of v "contain" all strategies of length ℓ' and

probing k vertices per turn, assuming that, initially, the target occupies a vertex in $L(pv)$. Let us say that v is *valid* if it contains at least one such winning strategy.

To find out if there is a winning strategy in G, let us proceed by dynamic programming, bottom-up from the leaves of this tree to the root. A leaf v of T is valid if and only if all of its child-edges would be labelled with a singleton (indeed, the leaves of T represent the last step of probing in a strategy of length ℓ, so the location of the target must be uniquely identified). A non-leaf vertex v is valid if and only if, for each of its unique child-edge labels, there exists a child that is valid (after a probing, there must be a winning strategy, whatever be the answer).

Therefore, there is a winning strategy for G if and only if the root is valid which can be decided in time $|V(T)| = n^{O(k\ell)}$. Indeed, the root r is trivially valid if at least one of its children is valid. □

3.1 When the number k of probed vertices per step is fixed

For a fixed integer $k \geq 1$, the k-Probe Localization problem takes a graph G and an integer $\ell \geq 1$ as inputs and asks whether $\lambda_k(G) \leq \ell$. Analogously, for any fixed integer $k \geq 2$, the k-Probe Relative-Localization problem takes a graph G and an integer $\ell \geq 1$ as inputs and asks whether $\lambda_k^{rel}(G) \leq \ell$.

Theorem 3.3. *For every $k \geq 1$, the k-Probe Localization problem is NP-complete in the class of graphs with a universal vertex.*

Proof. The problem is in NP by Claim 3.1. Let us prove it is NP-hard by a reduction from the 3-Dimensional Matching (3DM) problem which is a well known NP-hard problem. The 3DM problem takes a set $\mathcal{X} = I_1 \cup I_2 \cup I_3$ of $3n$ elements ($|I_1| = |I_2| = |I_3| = n$) and a set $\mathcal{S}$ of triples $(x, y, z) \in I_1 \times I_2 \times I_3$ as inputs and asks whether there are n triples of $\mathcal{S}$ that are pairwise disjoint.

Let $k \geq 1$ be a fixed integer and let $\mathcal{I} = (\mathcal{X}, \mathcal{S})$ be an instance of 3DM. First, we may assume that $|\mathcal{X}| = 3kn$ since, if not, it is sufficient to take k disjoint copies of $(\mathcal{X}, \mathcal{S})$. Moreover, we may assume that $m = |\mathcal{S}|$ is such that $2m - 1 \equiv 0 \mod k$ (for instance by adding dummy triples if needed). Let $\mathcal{X} = \{x_1, \ldots, x_{3kn}\}$ and $\mathcal{S} = \{S_1, \ldots, S_m\}$.

From $(\mathcal{X}, \mathcal{S})$, we construct, in polynomial time, a graph $G = (V, E)$ with the vertex-set $V = X \cup X'' \cup S \cup \{s\} \cup \{q\}$ such that (see Fig. 2):

- $X = X^1 \cup \cdots \cup X^{k+2}$ with $X^i = \{x_1^i, \ldots, x_{3kn}^i\}$ for every $i \leq k+2$. Each of the vertices x_j^i, for $i \in [\![1, k+2]\!]$, represents the element x_j, for $j \leq 3kn$;
- $X'' = \{x''_1, \ldots, x''_{(k+2)m}\}$;
- $S = S^1 \cup \cdots \cup S^{k+2}$ with $S^i = \{s_j^i, 1 \leq j \leq m\}$ for every $i \in [\![1, k+2]\!]$. Each of the vertices s_j^i, for $i \in [\![1, k+2]\!]$, represents the element S_j, for $j \leq m$.

The edges of G are as follows:

- there is an edge between s and every vertex of $V \setminus \{s\}$;
- there is an edge between q and every vertex of $X \cup X''$;
- for every $j \in [\![1, 3kn]\!]$ and every $g \in [\![1, m]\!]$ such that $x_j \in S_g$, there is an edge between x_j^i and s_g^i for every $i \in [\![1, k+2]\!]$.

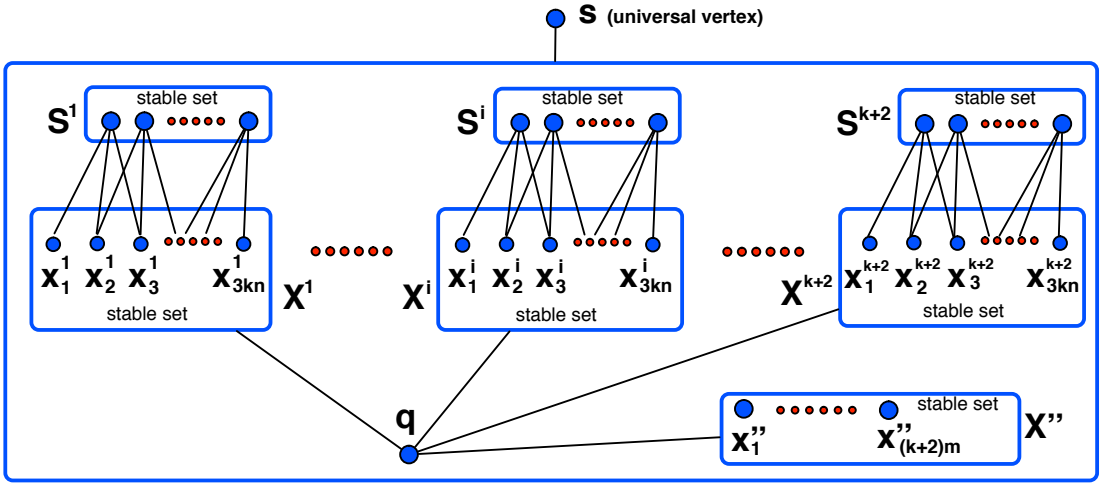

Figure 2: Example of a graph G constructed from an instance of 3DM in the proof of Theorem 3.3. A thin line between one vertex (blue circle) and one rectangle represents all edges between this vertex and every vertex in the rectangle. The instance of 3DM is encoded by the edges between the vertices in S^i (representing the sets) and the vertices in X^i, for every $1 \leq i \leq k+2$.

Let $p = \frac{m(k+2)-1}{k} \in \mathbb{N}$. We prove the theorem by showing that $\mathcal{I} = (\mathcal{X}, \mathcal{S})$ admits a 3DM if and only if $\lambda_k(G) \leq (k+2)n + p + 1$.

Claim 3.4. *If $\mathcal{I}$ admits a 3DM, then $\lambda_k(G) \leq (k+2)n + p + 1$.*

Proof of claim. Let $Y \subseteq \mathcal{S}$ be a 3DM of $\mathcal{I} = (\mathcal{X}, \mathcal{S})$ (of size $|Y| = kn$). Up to renumbering the sets and the elements, let us assume that $Y = \{S_1, S_2, \ldots, S_{kn}\}$ and assume that $S_i = \{x_{3(i-1)+1}, x_{3(i-1)+2}, x_{3(i-1)+3}\}$ for every $i \in [\![1, kn]\!]$. Note that, because Y is a 3DM of size kn, $\bigcup_{1 \leq i \leq kn} S_i = \mathcal{X}$ (*i.e.*, all elements are covered).

We describe a k-strategy Φ to locate the target in G in at most $(k+2)n + p + 1$ steps. The first step of Φ consists of probing only the vertex q. Three cases may occur. Either $H_1 = \{q\}$ (recall that H_s, here and further, denotes the set of vertices that remain possible locations for the target after the s^{th} step) in which case the target is located. Or the target is at distance 2 from q, *i.e.*, $H_1 = S$, in which case Φ sequentially probes every vertex of S but one until the target is located, which takes at most p extra steps. Or the target is at distance 1 from q and $H_1 = X \cup X'' \cup \{s\}$.

Hence, we may assume that $H_1 = X \cup X'' \cup \{s\}$. In this case, Φ proceeds by Phases of at most n steps each. There will be at most $k+2$ such Phases. Intuitively, during Phase $i \leq k+2$, the strategy Φ probes vertices in S^i in such a way that either the target is located at one of the vertices of X^i, or, at the end of the Phase, the target is known not to be in X^i.

Let us assume by induction on $1 \leq i \leq k+2$ and $1 \leq j \leq n$ that, before the j^{th} step of Phase i, if the target has not been located yet, then the set of possible locations for the target is

$$H_{1+(i-1)n+j-1} = \sigma \cup X'' \cup \left\{x^i_{3k(j-1)+1}, \ldots, x^i_{3kn}\right\} \cup \left(\bigcup_{i<y\leq k+2} X^y \right),$$

where $\sigma = \{s\}$ if $i = j = 1$ (or possibly, in the case $k = 1$, if $i = 1$ and $j = 2$), and $\sigma = \emptyset$ otherwise.

This holds for $i = j = 1$. Then, the strategy Φ consists of probing the vertices in $\mathcal{P}_{i,j} = \{s^i_{k(j-1)+1}, \ldots, s^i_{kj}\}$. There are three cases to consider. Before going into the details of the cases, recall that the sets $S_{k(j-1)+1}, \ldots, S_{kj}$ belong to the 3DM Y and so are pairwise disjoint. Hence, by construction of G, for every $a, b \in \mathcal{P}_{i,j}$, we have $(N_G(a) \cap X^i) \cap (N_G(b) \cap X^i) = \emptyset$.

- Either all vertices of $\mathcal{P}_{i,j}$ are at distance 1 from the target. In this case, the target is located at s (this case may only happen for $i = j = 1$ or, possibly, $i = 1$ and $j = 2$ in the case $k = 1$).

- Or exactly one vertex, say $s^i_{k(j-1)+x}$ for $1 \leq x \leq k$, of $\mathcal{P}_{i,j}$ is at distance 1 from the target. Let $y = k(j-1) + x$. In this case, the target must occupy one of $x^i_{3(y-1)+1}, x^i_{3(y-1)+2}, x^i_{3(y-1)+3}$ (the vertices corresponding to the elements that are contained in S_y). The strategy Φ probes two of these vertices, until the target is located in at most two extra steps. Therefore, in this case, the target is located in at most $1 + (i-1)n + j + 2 \leq (k+2)n + p + 1$ steps (since $i \leq k+2$ and $j \leq n$).

- The last case is when all the vertices of $\mathcal{P}_{i,j}$ are at distance 2 from the target. In particular the target cannot occupy a vertex in $U = \{s\} \cup \{x^i_{3k(j-1)+1}, \ldots, x^i_{3kj}\}$. And so, if $j < n$, then

$$H_{1+(i-1)n+j} = H_{1+(i-1)n+j-1} \setminus U = X'' \cup \left\{x^i_{3kj+1}, \ldots, x^i_{3kn}\right\} \cup \left(\bigcup_{i<y\leq k+2} X^y \right),$$

  hence the induction hypothesis holds for $j + 1$. Finally, if $j = n$, then

$$H_{1+in} = H_{1+(i-1)n+n-1} \setminus U = X'' \cup \left(\bigcup_{i+1<y\leq k+2} X^y \right)$$

  and the induction hypothesis holds for $i + 1$ and $j = 1$. In this case, Phase $i + 1$ starts if $i + 1 \leq k + 2$.

After the n^{th} step of Phase $k + 2$, we get that $H_{1+(k+2)n} = X''$. The strategy Φ ends by sequentially probing every vertex of X'' but one. So, the target can be located in at most p extra steps. Therefore, $\lambda_k(G) \leq (k+2)n + p + 1$. ◇

Claim 3.5. *If every 3DM of $\mathcal{I}$ has size strictly less than kn, then $\lambda_k(G) > (k+2)n+p+1$.*

Proof of claim. Let us assume that every 3DM of $\mathcal{I}$ has size strictly less than kn. We show that every k-strategy needs at least $(k+2)n + p + 2$ steps to guarantee the localization of the target in G. To avoid technicalities, let us assume that $H_0 = X \cup X''$, *i.e.*, the target is known *a priori* to occupy a vertex in $X \cup X''$. We show that even with this extra assumption (that is not favourable for the target), every k-strategy needs at least $(k+2)n + p + 2$ steps to guarantee the localization of the target.

Let Φ be any k-strategy. First, let us note that, since $H_0 = X \cup X''$ and both q and s are universal for $X \cup X''$, then probing q or s does not bring further information. Therefore, we may assume that Φ never probes q nor s. Let us now describe the information retrieved upon probing vertices in X, X'' or S.

(a) Let $u \in X''$. Note that $d_G(u,z) = 2$ for every $z \in X \cup X'' \setminus \{u\}$. Therefore, probing u only determines if the target is on u or not, and gives no further information. In other words, probing u only allows to remove u from the set of possible locations.

(b) Let $u \in X^i$ for any $i \leq k+2$. Note that $d_G(u,z) = 2$ for every $z \in X \cup X'' \setminus \{u\}$. Therefore, similarly, probing u only allows to remove u from the set of possible locations.

(c) Let $u \in S^i$ for any $i \leq k+2$. Let $\{x,y,z\} = N_G(u) \cap X^i$, *i.e.*, x,y,z are the vertices corresponding to the elements contained in the set that corresponds to u. Note that $d_G(u,z) = 2$ for every $z \in X \cup X'' \setminus \{x,y,z\}$. Therefore, probing u removes at most three vertices, namely x,y,z, from the set of possible locations.

(d) More generally, let $Z \subseteq S^i$ with $|Z| < kn$. Probing all vertices of Z allows to remove $N_G(Z) \cap X^i$, *i.e.*, at most $3|Z|$ vertices, from the set of possible locations.

(e) Finally, let $Z \subseteq S^i$ with $|Z| = kn$. Because $\mathcal{I}$ has no 3DM of size kn, there must be at least two vertices of Z whose neighbourhoods intersect in X^i. That is, $|N_G(Z) \cap X^i| \leq 3kn - 1$. Probing all vertices of Z allows to remove at most $3kn-1$ vertices from the set of possible locations.

Let $P \subseteq X \cup X'' \cup S$ be the set of all vertices that have been probed during the $(k+2)n+p+1$ first steps of Φ. We show that, at this point, the set of possible locations for the target still contains at least two vertices and so an extra step is required.

For every $0 \leq j \leq kn$, let α_j be the number of sets S^i that contain exactly $kn-j$ vertices of P. Formally, $\alpha_j = |\{i \mid 1 \leq i \leq k+2, |S^i \cap P| = kn-j\}|$. For every $kn < j \leq m$, let α_j be the number of sets S^i whose exactly j vertices have been probed, *i.e.*, $\alpha_j = |\{i \mid 1 \leq i \leq k+2, |S^i \cap P| = j\}|$. By definition, since $|S^i| = m$ for every $i \leq k+2$:

$$\sum_{0 \leq j \leq m} \alpha_j = k+2. \tag{1}$$

Let $y = |X \cap P|$ be the total number of vertices probed in X and let $y'' = |X'' \cap P|$ be the total number of vertices probed in X''. By definition of y, y'', and the α's, the total number ρ of vertices that have been probed after $(k+2)n+p+1$ steps satisfies:

$$\rho = y + y'' + \sum_{kn < j \leq m} j\alpha_j + \sum_{0 \leq j \leq kn} (kn-j)\alpha_j. \tag{2}$$

Moreover, since at most k vertices can be probed each step:

$$\rho \leq k[(k+2)n+p+1] \tag{3}$$

Note that, by Item (a) above, if $y'' \leq (k+2)m-2$, then at least two vertices have not been probed and, therefore, are still potential locations for the target (as noticed above, probing a vertex of X'' is the only way to remove it from the set of possible locations). In such a case, another step would be needed to ensure the localization. Therefore, we may assume that $y'' \in \{(k+2)m-1; (k+2)m\}$.

Let us assume that $y'' = (k+2)m$ (below, we point out the few differences in the case $y'' = (k+2)m-1$). In that case, all vertices in X'' are removed from the set of possible locations of the target that must be in X. Let $0 < j \leq kn$ and let $i \leq k+2$ such that $kn-j$ vertices have been probed in S^i. By Item (d) above, probing the vertices in S^i removes at

most $3(kn-j)$ vertices of X^i (and no other vertices) from the set of possible locations of the target. In other words, it leaves at least $3j$ vertices of X^i as possible locations. Let $i \leq k+2$ such that kn vertices have been probed in S^i. By Item (e) above, probing the vertices in S^i removes at most $3kn-1$ vertices of X^i (and no other vertices) from the possible locations of the target. In other words, one vertex of X^i is still a possible location.

Summing over all $j \in [\![0, kn]\!]$, the vertices probed in S leave at least $\alpha_0 + \sum_{1\leq j\leq kn} 3j\alpha_j$ vertices of X as possible locations for the target. To ensure the localization of the target without more steps, only one vertex of X must remain as a possible location (in the case when $y'' = (k+2)m-1$, *i.e.*, one vertex of X'' is still a possible location, then no vertex of X must remain possible). Since, by Item (b) above, only the y vertices probed in X may remove further vertices from the set of possible locations, it follows that:

$$y + 1 \geq \alpha_0 + \sum_{1\leq j\leq kn} 3j\alpha_j. \tag{4}$$

In the case where $y'' = (k+2)m-1$, this is $y \geq \alpha_0 + \sum_{1\leq j\leq kn} 3j\alpha_j$.

We are now ready to show that the above inequalities lead to a contradiction, proving that an extra step is required. For this purpose, let us consider again the total number ρ of vertices that have been probed during the first $(k+2)n+p+1$ steps.

$$\begin{aligned}
\rho &= y + y'' + \sum_{kn<j\leq m} j\alpha_j + \sum_{0\leq j\leq kn} (kn-j)\alpha_j && \textit{(Equation (2))}\\
&= y + y'' + \sum_{kn+1\leq j\leq m} (j-kn)\alpha_j + kn \sum_{0\leq j\leq m} \alpha_j - \sum_{0\leq j\leq kn} j\alpha_j \\
&= y + y'' + \sum_{kn+1\leq j\leq m} (j-kn)\alpha_j + kn(k+2) - \sum_{0\leq j\leq kn} j\alpha_j && \textit{(Equation (1))}\\
&= y + (k+2)m + \sum_{kn+1\leq j\leq m} (j-kn)\alpha_j + kn(k+2) - \sum_{0\leq j\leq kn} j\alpha_j && \textit{(if } y'' = (k+2)m\textit{)}\\
&\geq \alpha_0 + \sum_{1\leq j\leq kn} 3j\alpha_j - 1 + (k+2)m + \sum_{kn+1\leq j\leq m} (j-kn)\alpha_j + kn(k+2) - \sum_{0\leq j\leq kn} j\alpha_j \\
& && \textit{(Inequality (4))}\ \textit{(if } y'' = (k+2)m\textit{)}\\
&= \alpha_0 + \sum_{1\leq j\leq kn} 3j\alpha_j + pk + \sum_{kn+1\leq j\leq m} (j-kn)\alpha_j + kn(k+2) - \sum_{0\leq j\leq kn} j\alpha_j \ \textit{(by definition of } p\textit{)}\\
&= k[n(k+2)+p+1] + \sum_{kn+1\leq j\leq m} (j-kn)\alpha_j + \alpha_0 + \sum_{1\leq j\leq kn} 2j\alpha_j - k\\
&= k[n(k+2)+p+1] + 2(k+2) - 2\sum_{0\leq j\leq m} \alpha_j + \sum_{kn+1\leq j\leq m} (j-kn)\alpha_j + \alpha_0 + \sum_{1\leq j\leq kn} 2j\alpha_j - k \\
& && \textit{(Equation (1))}\\
&= k[n(k+2)+p+1] + 4 + \sum_{kn+1\leq j\leq m} (j-kn)\alpha_j - 2\sum_{kn+1\leq j\leq m} \alpha_j - \alpha_0 + \sum_{1\leq j\leq kn} 2(j-1)\alpha_j + k\\
&= k[n(k+2)+p+1] + 4 + \sum_{kn+2\leq j\leq m} (j-kn-1)\alpha_j - \sum_{kn+1\leq j\leq m} \alpha_j - \alpha_0 + \sum_{1\leq j\leq kn} 2(j-1)\alpha_j + k\\
&\geq k[n(k+2)+p+1] + 4 + k - \alpha_0 - \sum_{kn+1\leq j\leq m} \alpha_j\\
\rho &\geq k[n(k+2)+p+1] + 2 && \textit{(Equation (1))}
\end{aligned}$$

This contradicts Inequality (3) and concludes the proof of the claim. ◇ □

Via slight modifications, the previous proof can also be applied to prove the hardness of the k-Probe Relative-Localization problem.

Theorem 3.6. *For every $k \geq 2$, the k-Probe Relative-Localization problem is NP-complete in the class of graphs with a universal vertex.*

Proof. The proof of Theorem 3.3 applies, except that the strategy designed in Claim 3.4 has to start by probing both s and q (instead of only q). In this variant (with relative distances), the localization may require one more step (than with exact distances) in case the target is in $S \cup \{s\}$. The claim still holds since this case (the target in $S \cup \{s\}$) is not the worst case. Note that Claim 3.5 trivially holds for the variant with relative distances since $\lambda_k^{rel}(G) \geq \lambda_k(G)$ for all integers $k \geq 2$ and all graphs G. □

3.2 When the number ℓ of steps is fixed

For a fixed integer $\ell \geq 1$, the ℓ-Step Localization problem takes a graph G and an integer $k \geq 1$ as inputs and asks whether $\kappa_\ell(G) \leq k$. In the case where the target must be located through relative distances, the analogous problem ℓ-Step Relative-Localization is defined in the obvious way (but $k \geq 2$ in that case).

Theorem 3.7. *For every $\ell \geq 1$, the ℓ-Step Localization problem is NP-complete in the class of graphs with a universal vertex.*

Proof. For $\ell = 1$, the result follows from the fact that computing $\kappa_1(G)$ is exactly the same as computing the metric dimension $MD(G)$ of G, and that the problem of computing the metric dimension is NP-complete even when restricted to the class of graphs that contain a universal vertex [11]. So, from now on, let us assume that $\ell \geq 2$.

The problem is in NP by Claim 3.1. To prove the NP-hardness let us reduce the Metric Dimension problem (given a graph G and an integer $k \geq 1$, is $MD(G) \leq k$?) restricted to the class of graphs that contain a universal vertex, which is known to be NP-hard [11]. Let G be a graph that contains a universal vertex and k be an integer. We construct, in polynomial time, a graph G' such that $MD(G) \leq k$ if and only if a target hidden in G' can be located in at most ℓ steps by probing at most k vertices per step, *i.e.*, $\kappa_\ell(G') \leq k$.

The construction of G' is as follows. Start from $k(\ell-1)+1$ disjoint copies $G_1, \ldots, G_{k(\ell-1)+1}$ of G. Let v be a universal vertex of G, and for $1 \leq i \leq k(\ell-1)+1$, let v_i denote the copy of v in G_i. Finally, add a universal vertex u to the graph. This results in G'. Clearly, the construction is achieved in polynomial time.

We start by pointing out the following easy claim.

Claim 3.8. *For any $1 \leq a \leq k(\ell-1)+1$, if the target is known to occupy a vertex of G_a, then probing a vertex $w \in V(G' \setminus G_a)$ does not remove any vertex in G_a from the set of possible locations.*

Proof of claim. The vertex u is universal to G_a and, therefore, all vertices of G_a are the same distance from u and every shortest path from w to a vertex of G_a includes u. Thus, any two vertices of G_a cannot be distinguished via their distance to w. ◇

We now prove that $MD(G) \leq k$ if and only if $\kappa_\ell(G') \leq k$.

- First let us assume that $MD(G) \leq k$; we show that $\kappa_\ell(G') \leq k$. Consider the k-strategy where, during step s (for $1 \leq s \leq \ell - 1$), we probe the vertices in $\{v_{(s-1)k+1}, \ldots, v_{sk}\}$.

– If the target is at one of these vertices, say v_i, then it is located immediately at some step.

– If the target is at distance 1 from one probed vertex v_i, then it occupies a vertex in the corresponding G_i (unless $k = 1$, in which case the target could also occupy u). Note that, because G has diameter 2, then each of its copies G_i is an isometric subgraph of G'. Hence, any resolving set of size k of G (which exists since $MD(G) \leq k$) is also a resolving set for the vertices of G_i in G'. Probing such a resolving set in G_i during the next step then allows to locate the target. In the case $k = 1$, G_i has at most 3 vertices as otherwise, $MD(G) > 1$ since v_i is a universal vertex in G_i. Then, there are at most two other vertices in G_i that have not been probed (and are not adjacent if there are two, again since otherwise, $MD(G) > 1$), and thus, the target can be located in the next step by probing one of these vertices to distinguish it from u and the other.

– If the target is at distance 1 from all the $k \geq 2$ v_i vertices, then it is located at u.

– If at step $\ell - 1$ the target is at distance 2 from the probed vertices, then it is located in $G_{k(\ell-1)+1}$ and can be located at step ℓ since we have assumed that $MD(G) \leq k$ and each G_i is isometric in G'.

- Now we prove the other direction, that is, we show that $MD(G) > k$ implies that $\kappa_\ell(G') > k$. Since there are $k(\ell-1)+1$ copies of G_i and only $k(\ell-1)$ vertices can be probed during the first $\ell-1$ steps, then, on the last step, regardless of the employed strategy, there will always exist a copy, say G_a for some $1 \leq a \leq k(\ell-1)+1$, for which no vertices in G_a have been probed. If the target is hidden in G_a, then, by Claim 3.8, all the vertices of G_a are still potential locations for the target. The last step is then not sufficient to locate a target hidden in G_a since probing a vertex $w \in V(G' \setminus G_a)$ is useless by Claim 3.8, G_a is an isometric subgraph of G', and $MD(G_a) > k$. Hence, $\kappa_\ell(G') > k$. □

A proof establishing the hardness of ℓ-Step Relative-Localization can analogously be obtained by a reduction of the Centroidal Dimension problem.

Theorem 3.9. *For every $\ell \geq 1$, the ℓ-Step Relative-Localization problem is NP-complete in the class of graphs with a universal vertex.*

Proof. For $\ell = 1$, the result follows from the fact that $\kappa_1^{rel}(G)$ is exactly the centroidal dimension $CD(G)$ of G, and that computing the centroidal dimension is an NP-complete problem even when restricted to the class of graphs that contain a universal vertex [9]. So let $\ell \geq 2$ be fixed.

The problem is in NP by Claim 3.1. To prove its NP-hardness, let us reduce the Centroidal Dimension problem restricted to the class of graphs that contain a universal vertex, which is known to be NP-hard [9]. Let G be a graph that contains a universal vertex, and $k \geq 2$. We construct, in polynomial time, a graph G' with a universal vertex such that $CD(G) \leq k$ if and only if a target hidden in G' can be located in at most ℓ steps, by probing at most k vertices per step, *i.e.*, $\kappa_\ell^{rel}(G') \leq k$.

The construction of G' is as follows. Start from $k(\ell-1)+1$ disjoint copies $G_1, \ldots, G_{k(\ell-1)+1}$ of G. Let v be a universal vertex of G, and for $1 \leq i \leq k(\ell-1)+1$, let v_i denote the copy of v in G_i. Then, add all the edges so that $v_{k(\ell-1)+1}$ becomes a universal vertex in the whole resulting graph, which is G'.

Claim 3.10. *Let $1 \leq a \leq k(\ell-1)+1$, and assume the target is known to occupy, in G', any vertex of G_a. If $CD(G) > k$, then we cannot locate the target in one step by probing k vertices of G'.*

Proof of claim. Assume k vertices are probed in G_a. Since $CD(G_a) > k$ and G_a is an isometric subgraph of G', there exist at least two vertices $y_1, y_2 \in G_a$ that cannot be distinguished based on the information received. That is y_1 and y_2 have the same relative-distance vector. If any number of the k vertices probed in G_a had instead been replaced by vertices in $G' \setminus G_a$, then the relative-distance vectors of y_1 and y_2 may change but they would still be identical to one another since $v_{k(\ell-1)+1}$ is a universal vertex (and thus, distance 1 from both y_1 and y_2) and a cut vertex which separates all the G_i's. $\diamond$

We are now ready to prove that $CD(G) \leq k$ if and only if $\kappa_\ell^{rel}(G') \leq k$.

- First let us assume that $CD(G) \leq k$. We show that $\kappa_\ell^{rel}(G') \leq k$. Consider the k-strategy where, at step s for $1 \leq s \leq \ell-1$, we probe the vertices in $\{v_{(s-1)k+1}, \ldots, v_{sk}\}$. Then:
  - If the target is closer to one of the vertices in $\{v_{(s-1)k+1}, \ldots, v_{sk}\}$ probed at step s, say $v_{(s-1)k+x}$ for some integer $1 \leq x \leq k$, then the target is at a vertex in $G_{(s-1)k+x}$. Indeed, all the G_is are separated by a cut vertex $v_{k(\ell-1)+1}$ and since $v_{k(\ell-1)+1}$ is universal, it is equidistant from all the vertices of $\{v_{(s-1)k+1}, \ldots, v_{sk}\}$. Note that each G_i is an isometric subgraph of G'. Hence, any centroidal basis of size k of G (which exists since $CD(G) \leq k$) is also a centroidal basis for the vertices of G_i in G'. Probing such a centroidal basis in G_i allows to locate the target during the next step $s+1 \leq \ell$.
  - If the target is equidistant from each of the vertices in $\{v_{(s-1)k+1}, \ldots, v_{sk}\}$ probed at step s, then the target may not be at the vertices in $\{v_{(s-1)k+1}, \ldots, v_{sk}\}$ nor at the vertices of $G_{(s-1)k+1}, \ldots, G_{sk}$. Therefore, if $s < \ell-1$, then $H_s = \{v_{sk+1}, \ldots, v_{(s+1)k+1}\} \cup \bigcup_{0 \leq i \leq k} V(G_{sk+1+i})$. Hence, after $s = \ell-1$ steps, $H_s = V(G_{k(\ell-1)+1})$. Then, since each G_i is an isometric subgraph of G' and $CD(G) \leq k$, probing a centroidal basis in $G_{k(\ell-1)+1}$ allows to locate the target during the next step $s+1 = \ell$.
- Now we prove the other direction, that is, we show that $CD(G) > k$ implies that $\kappa_\ell^{rel}(G') > k$. Whatever be the probing strategy, if, on the last step, there exists a copy, say G_a for some $1 \leq a \leq k(\ell-1)+1$, for which no vertices in G_a have been probed, then there is no way to know at which vertex of G_a the target is located. Indeed, all G_i's are separated by a cut vertex, so probing a vertex in some G_i provides no information on any other G_j, $j \neq i$. Since there are $k(\ell-1)+1$ copies of G_i and only $k(\ell-1)$ vertices may be probed in the first $\ell-1$ steps, then, on the last step, regardless of the strategy, there will always exist a copy, say G_a for some $1 \leq a \leq k(\ell-1)+1$, for which no vertices in G_a have been probed. According to Claim 3.10, the last step is not sufficient to locate the target in G_a. Hence, $\kappa_\ell^{rel}(G') > k$. $\square$

4 The LOCALIZATION problem in trees

This section is devoted to the study of the LOCALIZATION problem in the class of trees. Recall that when $\ell = 1$, the problem is equivalent to the one of determining the metric dimension, which can easily be solved in polynomial time in trees [13, 20]. We first show

that when k and ℓ are part of the input, deciding whether $\lambda_k(T) \leq \ell$ for a given tree T is NP-complete. Our reduction actually shows that the difficulty of the problem comes from the choice of the nodes to be probed during the first step. Surprisingly, we show that the first step is actually the only source of hardness. More precisely, our main result is that if the first step is given (intuitively, either given by an oracle or imposed by an adversary), then an optimal strategy (according to this first pre-defined step) can be computed in polynomial time. As a consequence, we design a (+1)-approximation algorithm for the Localization problem in trees and prove that, in contrast with general graphs (Theorem 3.3), the k-Probe Localization problem is polynomial-time solvable in the class of trees.

4.1 NP-hardness of the first step

Before proceeding to the proof of the main result of this section, first recall the following observation.

Observation 4.1. *For every star S_n with n leaves, $\lambda_k(S_n) = \lceil \frac{n-1}{k} \rceil$.*

In particular, we first need to give an exact formula for λ_k for a particular class of trees before proving the main result of this section. More precisely, let $k \geq 1$ be fixed, and $1 < r \in \mathbb{N}$ be such that $r - 1 \equiv 0 \mod k$. For $1 < n \in \mathbb{N}$, we denote by S_n^r the tree obtained from r copies of S_n (the star with n leaves) by adding one new node c adjacent to the center of each of the r stars.

Lemma 4.2. *For every k, r, n as above,*

$$\lambda_k(S_n^r) = \frac{r-1}{k} + \left\lceil \frac{n-1}{k} \right\rceil.$$

Furthermore, $MD(S_n^r) = r(n-1)$.

Proof. For every $1 \leq i \leq r$ and $1 \leq j \leq n$, let c^i denote the center of the i^{th} copy of S_n, denoted by S^i, and let c_j^i denote the j^{th} leaf of the i^{th} copy of S_n. First, we prove that $\lambda_k(S_n^r) \leq \frac{r-1}{k} + \lceil \frac{n-1}{k} \rceil$. Consider the k-strategy Φ where, at each step $1 \leq s \leq \frac{r-1}{k}$, the nodes $c_1^{(s-1)k+1}, \ldots, c_1^{sk}$ are probed. If at step s, one of the probed nodes, say $c_1^{(s-1)k+x}$ for some $1 \leq x \leq k$, is:

- distance 0 from the target, then the target is located at $c_1^{(s-1)k+x}$;
- distance 1 from the target, then the target is located at $c^{(s-1)k+x}$;
- distance 2 from the target and $k = 1$, then the target is located at c or $c_y^{(s-1)k+x}$ for some $2 \leq y \leq n$. The target is then located in a total of at most $s + \lceil \frac{n-1}{k} \rceil$ steps since it occupies a leaf of the subgraph induced by $c^{(s-1)k+x}$ and all of its neighbours except for $c_1^{(s-1)k+x}$, which happens to be a star S_n that is also an isometric subgraph of S_n^r;
- distance 2 from the target and $k > 1$, then the target is located at c if it is also distance 2 from the other probed nodes. Otherwise, it is at $c_y^{(s-1)k+x}$ for some $2 \leq y \leq n$. The target is then located in a total of at most $s + \lceil \frac{n-2}{k} \rceil$ steps since it occupies a leaf of the subgraph induced by $c^{(s-1)k+x}$ and all of its neighbours except for c and $c_1^{(s-1)k+x}$, which happens to be a star S_{n-1} that is also an isometric subgraph of S_n^r.

If at step $s < \frac{r-1}{k}$ all of the probed nodes are at distance 3 from the target, then the target is located at one of the nodes $c^{sk+1}, \ldots, c^{(s+1)k}$. If at step $s < \frac{r-1}{k}$ all of the probed nodes are at distance 4 from the target, then the target is located at one of the nodes $c_j^{sk+1}, \ldots, c_j^{(s+1)k}$.

If at step $\frac{r-1}{k}$ all of the probed nodes are at distance 3 from the target, then the target is located at c^r. If at step $\frac{r-1}{k}$ all of the probed nodes are at distance 4 from the target, then the target is located at one of the nodes c_j^r. The target is then located in a total of at most $\frac{r-1}{k} + \lceil \frac{n-1}{k} \rceil$ steps since it occupies a leaf of the subgraph induced by c^r and all its neighbours except for c which happens to be a star S_n that is also an isometric subgraph of S_n^r.

We now prove that $\lambda_k(S_n^r) > \frac{r-1}{k} + \lceil \frac{n-1}{k} \rceil - 1$. We may assume that the target is on a leaf as this is not a favourable case for it. Consider a k-strategy. Since there are r copies of S_n in S_n^r and at most $k\frac{r-1}{k}$ nodes can be probed during the first $\frac{r-1}{k}$ steps, then, after step $\frac{r-1}{k}$, there will always exist a copy S^a for some $1 \leq a \leq r$ of S_n for which no nodes in S^a have been probed. Assuming the target is in S^a, note that the nodes probed in $S_n^r \setminus S^a$ during the previous steps did not provide any information on its location. Since S^a is a star with n leaves, we require at least $\lceil \frac{n-1}{k} \rceil$ additional steps to locate the target.

The last part of the statement, *i.e.*, $MD(S_n^r) = r(n-1)$, was proved *e.g.*, in [13, 20]. □

We are now ready to prove that the Localization problem remains NP-complete when restricted to trees.

Theorem 4.3. *The* Localization *problem is NP-complete in the class of trees.*

Proof. The problem is in NP by Claim 3.1. We now prove its NP-hardness by a reduction from the Hitting Set problem. The inputs are an integer $k \geq 1$, a ground set $B = \{b_1, \ldots, b_n\}$, and a set $\mathcal{S} = \{S_1, \ldots, S_m\}$ of subsets of B, *i.e.*, $S_i \subseteq B$ for every $i \leq m$. The Hitting Set problem aims at deciding if there exists a set $H \subseteq B$ such that $|H| \leq k$ and $H \cap S_i \neq \emptyset$ for every $i \leq m$.

Adding one new element to the ground set and adding this element to one single subset clearly does not change the solution. Therefore, by adding some dummy elements (each one belonging to a single subset), we may assume that all subsets are of the same size σ and that $\sigma - 1 \equiv 0 \mod k$.

Let γ be any integer such that $\gamma - 1 \equiv 0 \mod k$ and $\gamma > n - k - 1$. The instance T of the Localization problem is built as follows (see Fig. 3 for an illustration). Start with n node-disjoint paths $B_1, \ldots, B_n$ (called *branches*) of length $2m$, where $B_i = (b_1^i, \ldots, b_{2m+1}^i)$ for each $i \leq n$. Then add one new *root* node r adjacent to b_1^i for all $i \leq n$. For every $1 \leq j \leq m$ and for every $1 \leq i \leq n$ such that $b_i \in S_j$, add γ new nodes adjacent to b_{2j}^i. The subgraph induced by b_{2j}^i and by the γ leaves adjacent to it is referred to as the *star* representing the element i in the set S_j (or representing the set S_j in the branch i). The construction of T is clearly achieved in polynomial time.

Intuitively, it will always be favourable for the target to be located in a leaf of some star because γ is "huge". During the first step of any strategy, the *level* (roughly, the distance to the root) of the target can be identified. Each even level $2j$ corresponds to a set S_j. If, during the first step, one star corresponding to each even level can be eliminated from the possible locations (which corresponds to hit every subset), then the strategy finishes one step earlier than if all subsets cannot be hit (as, in such a situation, all stars would have to be checked).

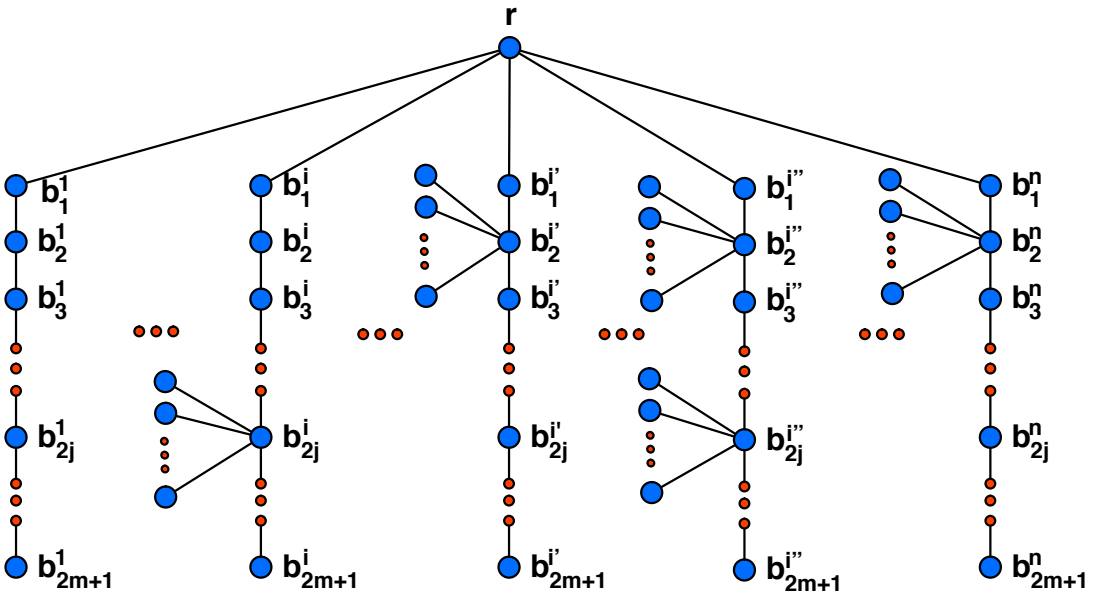

Figure 3: Example of a tree T constructed from an instance of Hitting Set in the proof of Theorem 4.3. In this example, the elements $b_{i'}, b_{i''}$, and b_n belong to the set S_1 (but not the elements b_1 and b_i) as figured by the three stars at level 2. The elements b_i and $b_{i''}$ belong to S_j (stars at level $2j$) but not the elements $b_1, b_{i'}$, and b_n.

More formally, we show below that $\lambda_k(T) \leq 1 + \frac{\sigma-1}{k} + \frac{\gamma-1}{k}$ if and only if there is a hitting set H of size at most k for $(B, \mathcal{S})$. Let us first show that if there is a hitting set H of size at most k for $(B, \mathcal{S})$, then $\lambda_k(T) \leq \ell$ for any $\ell \geq 1 + \frac{\sigma-1}{k} + \frac{\gamma-1}{k}$. W.l.o.g. (up to renumbering the elements), let us assume that $H = \{b_1, \ldots, b_k\}$ and let us present the corresponding k-strategy. During the first step, the nodes $b^1_{2m+1}, \ldots, b^k_{2m+1}$ are probed. We consider the following cases.

- First, if the target is at distance exactly $2m+1$ from one of (actually from all) the probed nodes, then it is located at r.
- Then, let us assume that the target is at distance strictly less than $2m+1$ from one of the probed nodes, w.l.o.g., that the target occupies a node in the branch B_1 (including the leaves of the stars in this branch). If the target is at odd distance from b^1_{2m+1}, then the target is located since there is a unique node at distance $2h+1$ from b^1_{2m+1} for each $0 \leq h \leq m$. Otherwise, the target is at distance $d = 2(m-h)$ from b^1_{2m+1} for some $0 \leq h < m$ (if $h = m$, then the target is trivially located). If $b_1 \notin S_{h+1}$, then b^1_{2m+2-d} is not the center of a star and b^1_{2m+1-d} is the unique node at distance d from b^1_{2m+1} and the target is located. Otherwise, the target may occupy b^1_{2m+1-d} or any leaf adjacent to b^1_{2m+2-d}. By Observation 4.1, this can be checked in $\lceil \frac{\gamma}{k} \rceil$ steps by sequentially checking each of these nodes but one. Overall, in this case, the target is located in at most $1 + \lceil \frac{\gamma}{k} \rceil$ steps (including the first one).
- Hence, we may assume that the target is at distance at least $2m+2$ from each of $b^1_{2m+1}, \ldots, b^k_{2m+1}$. Note that, in this case, the target is the same distance from every probed node. Said differently, the information brought by the first step is that the target is at some distance $d \geq 1$ from the root r and not in branches $B_1, \ldots, B_k$.

– If d is even, then the target can be at $b_d^{k+1}, \ldots, b_d^n$. Indeed, for every $i \leq n$, and any even distance d', there is a unique node at distance d' from r in the branch B_i. By Observation 4.1, the target can be located in $\lceil \frac{n-k-1}{k} \rceil$ steps by sequentially checking each of these nodes but one. Overall, it took $1 + \lceil \frac{n-k-1}{k} \rceil$ steps to locate the target.

– Otherwise, $d = 2j + 1$ for some $j \leq m$. Recall that H is a hitting set. In particular, $|S_j \setminus H| < |S_j| = \sigma$. In the worst case, $|S_j \setminus H| = \sigma - 1$ and, w.l.o.g. (up to renumbering), $S_j \setminus H = \{b_{k+1}, \ldots, b_{k+\sigma-1}\}$. In this case, the target can be located at $b_d^{k+1}, \ldots, b_d^n$ or at any leaf adjacent to one of the nodes $b_{2j}^{k+1}, \ldots, b_{2j}^{k+\sigma-1}$ (*i.e.*, the leaves of the stars corresponding to the set S_j in the branches that have not been hit). Then, the strategy continues by sequentially probing the nodes $b_d^{k+1}, \ldots, b_d^{n-1}$. Note that we start by the branches containing the stars that remain to be checked. There are two cases to be considered.

  * Either after checking $b_d^{k+1}, \ldots, b_d^{k+\sigma-1}$ in $\frac{\sigma-1}{k}$ steps (recall that $\sigma - 1 \equiv 0 \mod k$), the target is located to be in some star (this is the case if it is at distance 2 from one probed node). Then, it remains to identify which leaf of the star is the location of the target. This can be done in $\frac{\gamma-1}{k}$ steps by sequentially checking each of these leaves but one (Observation 4.1). Overall, in this case, the target has been located in $1 + \frac{\sigma-1}{k} + \frac{\gamma-1}{k}$ steps.
  * Or the target does not occupy a leaf of a star and is located after a total of $1 + \lceil \frac{n-k-1}{k} \rceil$ steps (including the first step).

To conclude, if the minimum size of a hitting set is at most k, then $\lambda_k(T) \leq \ell$ for any $\ell \geq 1 + \max\{\lceil \frac{\gamma}{k} \rceil, \lceil \frac{n-k-1}{k} \rceil, \frac{\sigma-1}{k} + \frac{\gamma-1}{k}\} = 1 + \frac{\sigma-1}{k} + \frac{\gamma-1}{k}$ (the last equality holds since $\gamma > n - k - 1$ and, since $\sigma - 1 \equiv 0 \mod k$ and $\sigma > 1$, we have $\frac{\sigma-1}{k} \geq 1$).

We now show that if there are no hitting sets of size at most k for $(B, \mathcal{S})$, then $\lambda_k(T) > \ell$ for any $\ell \leq 1 + \frac{\sigma-1}{k} + \frac{\gamma-1}{k}$. Consider any k-strategy. After the first step, at most k branches have some node that has been probed. These at most k branches correspond to at most k elements of the ground set B and, since all hitting sets of $(B, \mathcal{S})$ have size at least $k + 1$, there must be a set that does not contain any of these k elements. W.l.o.g., let $S_1 = \{b_1, \ldots, b_\sigma\}$ be this set. After the first step, let us assume that the target is located at distance 3 from the root (it is possible to decide this *a posteriori* since we are considering a worst case). Then, the target may be located at any leaf of some star corresponding to S_1. More precisely, the target may be at any node in $\{b_3^1, \ldots, b_3^\sigma\}$ or at any leaf adjacent to one of the nodes in $\{b_2^1, \ldots, b_2^\sigma\}$. Actually, the target may also be at other nodes (the third node of other branches), but we can ignore these choices. Even with this additional assumption, we show that the strategy will last for too long.

Indeed, after the first step, the instance becomes equivalent to an instance that consists of a rooted tree whose root has degree σ and each child of the root is adjacent to $\gamma+1$ leaves, and the target is known to occupy a leaf. By a direct adaptation of Lemma 4.2, locating the target takes another $\frac{\sigma-1}{k} + \lceil \frac{\gamma}{k} \rceil$ steps. Overall, locating the target thus requires at least $1 + \frac{\sigma-1}{k} + \lceil \frac{\gamma}{k} \rceil$ steps. Since $\gamma - 1 \equiv 0 \mod k$, then $\lceil \frac{\gamma}{k} \rceil > \frac{\gamma-1}{k}$ and $\lambda_k(T) > 1 + \frac{\sigma-1}{k} + \frac{\gamma-1}{k}$. □

4.2 A polynomial-time algorithm for the next steps

The proof of Theorem 4.3 shows that, in our reduction, choosing the nodes to be probed during the first step to ensure an optimal strategy is equivalent to finding a minimum hitting set. We show here that this first step is actually the only source of hardness for solving Localization in trees.

The key argument is the following easy remark. Let us consider a tree T where an immobile target is hidden and assume that a single node $r \in V(T)$ is probed. After this single probe, the distance $d \in \mathbb{N}$ between the target and r is revealed. Therefore, from the second step, the instance becomes equivalent to a tree T' (a subtree of T) rooted in r, whose leaves (all of them) are the same distance d from r, and where the target is known to occupy some leaf of T'. We first present an algorithm that computes in polynomial time (independent of k and ℓ) an optimal strategy to locate the target in such instances.

Let $\mathcal{T}$ be the set of rooted trees with all leaves the same distance from the root. Given a rooted tree $(T, r) \in \mathcal{T}$ (in what follows, we omit r when it is clear from the context), let $\lambda_k^L(T)$ be the minimum integer h such that there exists a k-strategy Φ for locating a target in at most h steps knowing *a priori* that the target occupies some leaf of T. The next claim is one of the key arguments that makes the problem easier in this context. For any node v in a rooted tree (T, r), we denote by T_v the subtree rooted at v.

Claim 4.4. *Let $(T, r) \in \mathcal{T}$ be a tree rooted in r and v be a child of r. If the target is known to occupy a leaf of T, then probing any node in T_v allows to learn if the target occupies a leaf of T_v or a leaf of $T \setminus T_v$.*

Proof of claim. Let d be the distance between r and the leaves of T. Let w be any node of T_v and let d' be the distance between w and r. The claim follows from the fact that the target occupies a leaf of T_v if and only if its distance to w is strictly less than $d + d'$. $\diamond$

Let $T \in \mathcal{T}$ be a tree rooted in r and v be a child of r, and let us assume that the secret location of the target is some leaf of T_v. Note that $(T_v, v) \in \mathcal{T}$. Let us assume that T_v is not a path and let s be the first step of an optimal strategy Φ in T that probes some node of T_v (such a step s must exist since otherwise the target would never be located in T_v). By Claim 4.4, it is sufficient to probe a single node of T_v to learn whether the target occupies a leaf of T_v. Then, applying an optimal strategy ϕ_v in T_v will locate the target in a total of $s + \lambda_k^L(T_v) - 1$ steps if the first step of ϕ_v only requires probing a single vertex of T_v and $s + \lambda_k^L(T_v)$ steps otherwise. So, it may be possible to do better. Indeed, probing several nodes of T_v during the s^{th} step of Φ may serve not only to locate the target in T_v but also to "play" the first step of Φ_v. Doing so, the strategy will take only $s + \lambda_k^L(T_v) - 1$ steps. Let $v_1, \ldots, v_{d^*}$ denote the children of r. So, elaborating, an optimal strategy will consist of doing a tradeoff between probing one single node in several of the T_{v_i}'s (and locating "quickly" in which subtree T_{v_i} the target is hidden, since several of them are considered simultaneously) and probing more nodes in some of the T_{v_i}'s in order to get a head start for the strategy in the case the T_{v_i} hosting the target is identified.

For any tree T, let $\pi(T)$ be the minimum integer $q \leq k$ such that there exists a k-strategy for locating a target in T in at most $\lambda_k^L(T)$ steps, knowing *a priori* that the target occupies some leaf of T, and such that q nodes are probed during the first step.

To illustrate this need of a tradeoff and the importance of π, let us consider the example depicted in Fig. 4. The root r of T has eight children $v_1, \ldots, v_8$ with the pairs $(\lambda_k^L(T_{v_i}), \pi(T_{v_i}))$ being $(4,2)$, $(4,1)$, $(3,3)$, $(3,3)$, $(2,2)$, $(2,2)$, $(1,1)$, and $(0,0)$, respectively. Let $k = 4$. Here, the target can be located in at most four steps, through the following strategy.

- Step 1. The probed nodes are those labeled ① in Fig. 4, that is, two nodes of T_{v_1}, one node of T_{v_2}, and one node of T_{v_3}. If the target occupies some leaf of T_{v_1} or T_{v_2}, then there is a strategy for locating the target in at most $\lambda_k^L(T_{v_1}) - 1 = \lambda_k^L(T_{v_2}) - 1 = 3$ extra steps because $\pi(T_{v_1})$ ($\pi(T_{v_2})$, resp.) nodes of T_{v_1} (T_{v_2}, resp.) have been probed. If the target occupies some leaf of T_{v_3}, then there is a strategy for locating the target

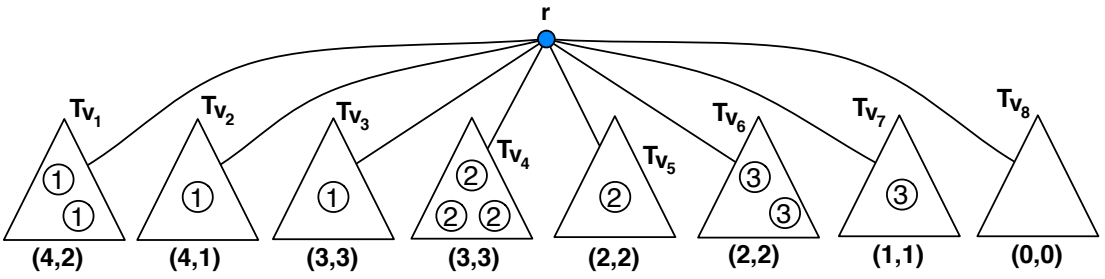

Figure 4: A tree $(T, r) \in \mathcal{T}$ rooted at r. The eight children of r are $v_1, \ldots, v_8$. The pair $(\lambda_k^L(T_{v_i}), \pi(T_{v_i}))$ for each T_{v_i} is written below the corresponding subtree. In the figure, one (two, three, resp.) ⓧ in a subtree corresponds to one (two, three, resp.) node (nodes) of this subtree being probed during step x.

Algorithm 1 $\mathcal{A}_1(k, (T, r))$

Require: An integer k and a tree $(T, r) \in \mathcal{T}$ rooted in r with children $v_1, \ldots, v_{d^*}$
Ensure: $(\lambda_k^L(T), \pi(T))$
1: **if** (T, r) is a rooted path **then**
2: **return** $(0, 0)$
3: **for** $i = 1$ to d^* **do**
4: Let $(\lambda_i, \pi_i) = \mathcal{A}_1(k, (T[i], v_i))$ *// recursive calls to Algorithm 1*
5: Sort the $(\lambda_i, \pi_i)_{1 \leq i \leq d^*}$ in non-increasing lexicographical order
6: **return** $\mathcal{A}_2(k, (T, r), (\lambda_i, \pi_i)_{1 \leq i \leq d^*})$ *// call to Algorithm 2*

in at most $\lambda_k^L(T_{v_3}) = 3$ extra steps (that is a total of four steps). Thus, assume that the target occupies a leaf of some subtree T_{v_i}, $4 \leq i \leq 8$.

- Step 2. The probed nodes are those labeled ② in Fig. 4, that is, three nodes of T_{v_4} and one node of T_{v_5}. If the target occupies some leaf of T_{v_4} or T_{v_5}, then using similar arguments to those above, we can show there is a strategy for locating the target in at most two extra steps (that is a total of four steps). Thus, assume that the target occupies a leaf of T_{v_6}, T_{v_7} or T_{v_8}.

- Step 3. The probed nodes are those labeled ③ in Fig. 4, that is, two nodes of T_{v_6} and one node of T_{v_7}. Again, if the target occupies some leaf of T_{v_6} or T_{v_7}, then, with at most one extra step, the target is located. Otherwise, the target is on T_{v_8} and there is no need for an extra step.

Let $(T, r) \in \mathcal{T}$ be a tree rooted in r and let $v_1, \ldots, v_{d^*}$ be the children of r. From previous arguments, the computation of an optimal strategy for T consists of determining, for each subtree T_{v_i} ($1 \leq i \leq d^*$), the first step for which a node of T_{v_i} will be probed (if the target has not been located in a different subtree at a previous step). If one node is probed during this step, then $\lambda_k^L(T_{v_i})$ extra steps are needed if the target occupies some leaf of T_{v_i} (unless $\pi(T_{v_i}) = 1$ in which case $\lambda_k^L(T_{v_i}) - 1$ extra steps are needed). Furthermore, if we want to locate the target in at most $\lambda_k^L(T_{v_i}) - 1$ extra steps (if the target occupies some leaf of T_{v_i}), then $\pi(T_{v_i})$ nodes of T_{v_i} must be probed during this step. Algorithms 1 and 2 compute such an optimal strategy for a tree in $\mathcal{T}$ in polynomial time. We first describe their behaviour, before focusing on proving their correctness.

Algorithm 2 $\mathcal{A}_2(k,(T,r),(\Lambda,\Pi))$

Require: An integer k and a tree $(T,r)\in\mathcal{T}$ rooted in r with $v_1,\dots,v_{d^*}$ such that $(\Lambda,\Pi)=(\lambda_i,\pi_i)_{1\leq i\leq d^*}$ is sorted in non-increasing lexicographical order

Ensure: $(\lambda_k^L(T),\pi(T))$

1: $\ell\leftarrow 1$, $p\leftarrow k$, $d\leftarrow d^*$
2: **if** $T[d^*]$ is a rooted path **then**
3: $\quad d\leftarrow z$ where $0\leq z<d^*$ is the smallest integer such that $T[z+1]$ is a rooted path
4: $\quad \ell\leftarrow 1+\left\lceil\frac{d^*-d-1}{k}\right\rceil$ // $\ell\leftarrow 1+\lambda_k^L(T[d+1,d^*])$ *(Lem. 4.5)*
5: $\quad p\leftarrow k+k\left(\left\lceil\frac{d^*-d-1}{k}\right\rceil-\left\lceil\frac{d^*-d-1}{d^*-d}\right\rceil\right)-(d^*-d-1)$ // $p\leftarrow k-\pi(T[d+1,d^*])$ *(Lem. 4.5)*
6: **for** $i=d$ down to 1 **do**
7: $\quad$ **if** $p=0$ or $\ell<\lambda_i+1$ **then**
8: $\quad\quad p\leftarrow k$, $\ell\leftarrow\max(\ell+1,\lambda_i+1)$
9: $\quad \alpha\leftarrow\pi_i-(\pi_i-1)\lceil(\ell-(\lambda_i+1))/\ell\rceil$ // $\alpha=\pi_i$ *if, in Line 7,* $\ell<\lambda_i+1$*, and* $\alpha=1$ *otherwise.*
10: $\quad$ **if** $\alpha\leq p$ **then**
11: $\quad\quad p\leftarrow p-\alpha$
12: $\quad$ **else**
13: $\quad\quad p\leftarrow k-1$, $\ell\leftarrow\ell+1$ // $\ell=1+\lambda_k^L(T[i,d^*])$; $p=k-\pi(T[i,d^*])$ *(Lem. 4.9)*
14: **return** $(\ell-1,k-p)$

Description of Algorithm 1 The main algorithm $\mathcal{A}_1(k,(T,r))$ takes an integer $k\geq 1$ and a rooted tree $(T,r)\in\mathcal{T}$ as inputs and computes $(\lambda_k^L(T),\pi(T))$ and a corresponding k-strategy. It proceeds bottom-up by dynamic programming from the leaves to the root. Precisely, let $v_1,\dots,v_{d^*}$ be the children of r. For any $1\leq i\leq j\leq d^*$, let $T[i]=T_{v_i}$ be the subtree rooted at v_i, and let $T[i,j]=\{r\}\cup T_{v_i}\cup\ldots\cup T_{v_j}$ ($T[i,j]=\emptyset$ if $i>j$). To lighten the notations, let us set $\lambda_i=\lambda_k^L(T[i])$ and $\pi_i=\pi(T[i])$ for every $1\leq i\leq d^*$. Assume that, $(\Lambda,\Pi)=(\lambda_i,\pi_i)_{1\leq i\leq d^*}$ have been computed recursively and sorted in non-increasing lexicographical order. Then, $\mathcal{A}_2(k,(T,r),(\Lambda,\Pi))$, described in Algorithm 2, takes the integer $k\geq 1$, the rooted tree $(T,r)\in\mathcal{T}$, and the sorted tuple (Λ,Π) as inputs and computes $(\lambda_k^L(T),\pi(T))$ and a corresponding strategy.

Description of Algorithm 2 We now informally describe $\mathcal{A}_2(k,(T,r),(\Lambda,\Pi))$. The first part, from Lines 2 to 5, deals with the subtrees $T_{v_{d+1}},\dots,T_{v_{d^*}}$ that are rooted paths (T_{v_i}'s being paths rooted at one of their two ends, while the second end is a leaf). In other words, these Lines deal with the T_{v_i}'s such that $(\lambda_i,\pi_i)=(0,0)$. Indeed, this case is somehow pathologic, and needs to be treated separately.

The second part (from Line 6) of Algorithm 2 goes as follows. Informally, $\mathcal{A}_2(k,(T,r),(\Lambda,\Pi))$ recursively builds, for $i=d$ down to 1, an optimal k-strategy Φ for $T[i,d^*]$ from an optimal k-strategy Φ' of $T[i+1,d^*]$ and from an optimal k-strategy Φ'' of $T[i]$ (the latter one being given as input through (λ_i,π_i)). In other words, $(\lambda_k^L(T[i,d^*]),\pi(T[i,d^*]))$ is computed from $(\lambda_k^L(T[i+1,d^*]),\pi(T[i+1,d^*]))$, and (λ_i,π_i). For every $1\leq i\leq d+1$, let ℓ_i (resp., p_i) denote the value of ℓ (resp. of p) just before the $(d+2-i)^{\text{th}}$ iteration of the **for** loop (so, ℓ_1 and p_1 are the final values of ℓ and p). Intuitively, let us assume that an optimal strategy for $T[i+1,d^*]$ has been computed, that it takes at most $\ell_{i+1}-1$ steps, and that it requires a minimum of $k-p_{i+1}=\pi(T[i+1,d^*])$ nodes to be probed during its first step. One can see p_{i+1} as the left-over budget from the previous step that can be used "for free" (without increasing ℓ) towards probing vertices in $T[i]$. Roughly, there are five cases to consider.

- If $\pi_i \leq p_{i+1}$ and $\lambda_i = \ell_{i+1} - 1$, then the strategy Φ follows Φ' but, in addition, probes π_i nodes of $T[i]$ during its first step. If the target is in $T[i]$, then Φ follows Φ'' (and takes a total of at most λ_i steps), otherwise, it proceeds as Φ' (and takes a total of at most $\ell_{i+1} - 1$ steps). We get $\ell_i = \ell_{i+1}$ and $p_i = p_{i+1} - \pi_i$.

- Else, if $\pi_i > p_{i+1} > 0$ and $\lambda_i = \ell_{i+1} - 1$, then the first step of Φ probes a unique node in $T[i]$. If the target is in $T[i]$, then Φ follows Φ'' (and takes a total of at most $\lambda_i + 1$ steps). Otherwise, it proceeds as Φ' (and takes a total of at most ℓ_{i+1} steps). We get $\ell_i = \ell_{i+1} + 1$ and $p_i = k - 1$.

- Else, if $p_{i+1} = 0$ and $\lambda_i \leq \ell_{i+1} - 1$, then the first step of Φ probes a unique node in $T[i]$. If the target is in $T[i]$, then Φ follows Φ'' (and takes a total of at most $\lambda_i + 1$ steps). Otherwise, it proceeds as Φ' (and takes a total of at most ℓ_{i+1} steps). We get $\ell_i = \ell_{i+1} + 1$ and $p_i = k - 1$.

- Else, if $\lambda_i < \ell_{i+1} - 1$ and $p_{i+1} > 0$, then the strategy Φ follows Φ' but, in addition, probes one node of $T[i]$ during its first step. If the target is in $T[i]$, then Φ follows Φ'' (and takes a total of at most $\lambda_i + 1$ steps), otherwise, it proceeds as Φ' (and takes a total of at most $\ell_{i+1} - 1$ steps). We get $\ell_i = \ell_{i+1}$ and $p_i = p_{i+1} - 1$.

- Finally, if $(\lambda_i > \ell_{i+1} - 1)$, then the strategy Φ probes π_i nodes in $T[i]$ during the first step. If the target is in $T[i]$, then Φ follows Φ'' (and takes a total of at most λ_i steps), otherwise, it proceeds as Φ' (and takes a total of at most ℓ_{i+1} steps). We get $\ell_i = \lambda_i + 1$ and $p_i = k - \pi_i$.

Correctness and complexity of Algorithms 1 and 2 We start by proving the correctness of the two main parts of Algorithm 2, *i.e.*, that of the peculiar case (Lines 2 to 5) and of the general case (from Line 6).

First, we consider the first part of Algorithm 2. Lemma 4.5 below proves that Lines 2 to 5 compute $(\lambda_k^L(T[v_{d+1}, d^*]), \pi(T[v_{d+1}, d^*]))$. Let us recall Observation 4.1, which states that, for every star S_n with n leaves, $\lambda_k(S_n) = \lceil \frac{n-1}{k} \rceil$.

We define $\mathcal{S} \subset \mathcal{T}$ as the set of subdivided rooted stars S (*i.e.*, trees with at most one node of degree at least 3) with all leaves the same distance from the root, where the root of S is the (unique) node with degree at least 3 or one of the two ends of S is a path.

Lemma 4.5. *For every subdivided rooted star $S \in \mathcal{S}$ with d leaves, $\lambda_k^L(S) = \lceil \frac{d-1}{k} \rceil$ and $\pi(S) = -k(\lceil \frac{d-1}{k} \rceil - \lceil \frac{d-1}{d} \rceil) + (d-1)$.*

Proof. The strategy consists of sequentially probing each leaf of S but one. Either the target will be probed at some step, or it must be in the unique leaf that has not been probed. During the first step, $\pi(S)$ leaves are probed, and exactly k leaves are probed during every other step. Such a strategy lasts for $\lceil \frac{d-1}{k} \rceil$ steps.

For any strategy using less than $\lceil \frac{d-1}{k} \rceil$ steps, the nodes of at most $k(\lceil \frac{d-1}{k} \rceil - 1) \leq d-2$ branches have been probed. Hence, there are at least two branches of S for which no nodes have been probed and so it is not possible to decide which one of these branches is occupied by the target.

Similarly, it can be checked that, for any strategy using at most $\lceil \frac{d-1}{k} \rceil$ steps and probing less than $\pi(S)$ nodes during the first step, there are at least two branches of S for which no nodes have been probed. To be convinced of that point, notice that $\pi(S) = -k(\lceil \frac{d-1}{k} \rceil - \lceil \frac{d-1}{d} \rceil) + (d-1)$ is equivalent to:

- $\pi(S) = 0$ if $d - 1 = 0$;

- $\pi(S) = k$ if $d - 1 > 0$ and $(d-1) \mod k = 0$;
- $\pi(S) = (d-1) \mod k$ otherwise.

This concludes the proof. □

We now focus on proving the correctness of the second part of Algorithm 2, which is mainly done in Lemma 4.9 below. We first introduce three easy observations, whose proofs are omitted.

Observation 4.6. *Let $(T, r) \in \mathcal{T}$ be a rooted tree. Then, $\lambda_k^L(T) = 0$ if and only if T is a rooted path, and $\pi(T) = 0$ if and only if T is a rooted path.*

Although the following observation (closedness of λ_k under subtree) does not hold in general graphs (see Observation 2.1), it can easily be seen that this holds in the case of trees.

Observation 4.7. *For any tree T and any subtree T' of T, $\lambda_k^L(T') \leq \lambda_k(T') \leq \lambda_k(T)$ and $\lambda_k^L(T') \leq \lambda_k^L(T)$.*

The next observation is obvious (indeed, to prove it, just note that the first step probing a single vertex can simply be ignored) but will be quite useful in what follows.

Observation 4.8. *For any tree T, there exists a k-strategy for locating the target in at most $\lambda_k(T) + 1$ steps (resp., in at most $\lambda_k^L(T) + 1$ steps if the target is known to occupy a leaf) and that probes a single arbitrary node during its first step.*

We are now ready to prove the next result, which essentially proves that the second part of Algorithm 2 is correct. That is, we prove, provided the T_{v_i}'s are sorted in non-increasing lexicographical order (over (λ_i, π_i)), that the strategy Φ described earlier is optimal for $T[i, d^*]$, that is, it computes $(\lambda_k^L(T[i, d^*]), \pi(T[i, d^*]))$.

Lemma 4.9. *For every $1 \leq i \leq d+1$, we have $\lambda_k^L(T[i, d^*]) = l_i - 1$ and $\pi(T[i, d^*]) = k - p_i$.*

Proof. The proof is by induction on $d + 1 - i \leq d + 1$. For $i = d + 1$, there are two cases to be considered.

- If $d = d^*$ (*i.e.*, the condition on Line 2 is not satisfied), then, before the first iteration, $\ell_{d+1} = 1, p_{d+1} = k$ and $T[d+1, d^*] = \emptyset$, and so $\lambda_k^L(\emptyset) = \ell_{d+1} - 1 = 0$ and $\pi(\emptyset) = k - p_{d+1} = 0$. So the induction hypothesis is satisfied for $i = d + 1$.
- Otherwise, $d < d^*$ and $T_{v_{d+1}}, \ldots, T_{v_{d^*}}$ are rooted paths. That is, $T[d+1, d^*] \in \mathcal{S}$. Then, the induction hypothesis for $i = d + 1$ is satisfied by Lemma 4.5 and Lines 2 to 5 of Algorithm 2.

Let us assume that the induction hypothesis holds for $1 < i + 1 \leq d + 1$. That is, at the end of the $(d-i)^{\text{th}}$ iteration of the **for** loop, $\lambda_k^L(T[i+1, d^*]) = \ell_{i+1} - 1$ and $\pi(T[i+1, d^*]) = k - p_{i+1}$. We will prove that it is also true after the next iteration of the **for** loop, *i.e.*, $\lambda_k^L(T[i, d^*]) = \ell_i - 1$ and $\pi(T[i, d^*]) = k - p_i$.

It is very important to note that Lines 2 and 3 imply that $\lambda_i > 0$ and $\pi_i > 0$, for every $1 \leq i \leq d$. We consider five cases depending on the values of $p_{i+1}, \pi_i, \lambda_i$, and ℓ_{i+1}.

- Case $0 < \pi_i \leq p_{i+1}$, $\ell_{i+1} = \lambda_i + 1$.

  By the induction hypothesis, $\lambda_k^L(T[i+1, d^*]) = \ell_{i+1} - 1 = \lambda_i$ and $\pi(T[i+1, d^*]) = k - p_{i+1}$. Because the value of ℓ at the beginning of this iteration of the **for** loop is $\ell_{i+1} = \lambda_i + 1$, then $\alpha = \pi_i$. Then, since $\pi_i \leq p_{i+1}$, we get that p becomes $p - \alpha = p_{i+1} - \pi_i$ and ℓ is not modified. Hence, $\ell_i = \ell_{i+1}$ and $p_i = p_{i+1} - \pi_i$.

  We now prove that $\lambda_k^L(T[i, d^*]) = \ell_{i+1} - 1$ and $\pi(T[i, d^*]) = k - p_{i+1} + \pi_i$. By Observation 4.7, we have $\lambda_k^L(T[i, d^*]) \geq \lambda_k^L(T[i+1, d^*]) = \ell_{i+1} - 1$. To prove that $\lambda_k^L(T[i, d^*]) \leq \lambda_k^L(T[i+1, d^*]) = \ell_{i+1} - 1$, it is sufficient to describe a strategy Φ for $\lambda_k^L(T[i, d^*])$ with a total of at most $\ell_{i+1} - 1$ steps.

  Let Φ' be an optimal strategy for $T[i+1, d^*]$ probing at most $\pi(T[i+1, d^*])$ nodes during the first step. Also, let Φ'' be an optimal strategy for $T[i]$ probing at most π_i nodes during the first step. The first step of Φ consists of probing π_i nodes of $T[i]$ (as Φ'') and $\pi(T[i+1, d^*]) = k - p_{i+1}$ nodes of $T[i+1, d^*]$ (as Φ'). By assumption, $\pi_i \leq p_{i+1}$, and, by the induction hypothesis, $\pi(T[i+1, d^*]) = k - p_{i+1}$, so $\pi_i + \pi(T[i+1, d^*]) \leq k$ and at most k nodes are probed. By Claim 4.4, this first step allows to decide if the target is in $T[i]$ or not (in the latter case, it is in $T[i+1, d^*]$). If the target is in $T[i]$, then Φ continues by following the strategy Φ'' in $T[i]$, which will locate the target in at most $\lambda_i - 1 = \ell_{i+1} - 2$ extra steps. Otherwise (the target is in $T[i+1, d^*]$), Φ continues by following the optimal strategy Φ' for $T[i+1, d^*]$ which will locate the target in at most $\lambda_k^L(T[i+1, d^*]) - 1 = \ell_{i+1} - 2$ extra steps. In all cases, Φ locates the target in at most $\ell_{i+1} - 1$ steps.

  We now prove that $\pi(T[i, d^*]) = k - p_{i+1} + \pi_i$. For purpose of contradiction, let us assume that there is a strategy for locating the target in $T[i, d^*]$ in at most $\lambda_i = \ell_{i+1} - 1$ steps and probing strictly less than $k - p_{i+1} + \pi_i$ nodes during the first step. By definition, at least π_i nodes of $T[i]$ must be probed during the first step to locate the target in at most $\lambda_i = \ell_{i+1} - 1$ steps. Thus, it means that strictly less than $k - p_{i+1}$ nodes of $T[i+1, d^*]$ can be probed during the first step. This contradicts that the strategy performs in at most $\lambda_i = \ell_{i+1} - 1$ steps since $\pi(T[i+1, d^*]) = k - p_{i+1}$.

- Case $\pi_i > p_{i+1} > 0$, $\ell_{i+1} = \lambda_i + 1$.

  In this case, it can be checked that $\alpha = \pi_i$ and that the **else** instruction (Line 12) is executed and so $\ell_i = \ell_{i+1} + 1$ and $p_i = k - 1$. We will prove that $\lambda_k^L(T[i, d^*]) = \ell_{i+1}$ and $\pi(T[i, d^*]) = 1$.

  By the induction hypothesis, $\lambda_k^L(T[i+1, d^*]) = \ell_{i+1} - 1 = \lambda_i$ and $\pi(T[i+1, d^*]) = k - p_{i+1}$.

  We prove that $\lambda_k^L(T[i, d^*]) \geq \lambda_k^L(T[i+1, d^*]) + 1 = \ell_{i+1}$. For purpose of contradiction, let us assume that $\lambda_k^L(T[i, d^*]) < \ell_{i+1}$ and let Φ' be a strategy for locating the target in $T[i, d^*]$ in at most $l_{i+1} - 1$ steps. Since $\ell_{i+1} - 1 = \lambda_i$, then at least π_i nodes of $T[i]$ must be probed during the first step. Since $\lambda_k^L(T[i+1, d^*]) = \ell_{i+1} - 1 = \lambda_i$ and $\pi(T[i+1, d^*]) = k - p_{i+1}$, at least $k - p_{i+1}$ nodes of $T[i+1, d^*]$ must be probed during the first step. This means that at least $\pi_i + k - p_{i+1} > k$ nodes must be probed during the first step, a contradiction.

  We now prove that $\lambda_k^L(T[i, d^*]) = \ell_{i+1}$. It is sufficient to design a strategy Φ for locating the target in $T[i, d^*]$ in at most ℓ_{i+1} steps. By Observation 4.8, there is a strategy Φ' for $T[i]$ for locating the target in at most $\lambda_i + 1$ steps that probes a single node during the first step. Also, let Φ'' be an optimal strategy for $T[i+1, d^*]$. The first step of Φ consists of probing one node of $T[i]$. If the target is in $T[i]$, the strategy

continues with Φ' (in at most $\lambda_i = \ell_{i+1} - 1$ steps), otherwise, the strategy continues with Φ'' (in at most $\lambda_k^L(T[i+1, d^*]) = \ell_{i+1} - 1$ steps). From this, we deduce that $\pi(T[i, d^*]) \leq 1$ and, by definition of π, we get that $\pi(T[i, d^*]) = 1$.

- Case $p_{i+1} = 0$, $\ell_{i+1} \geq \lambda_i + 1$.

  In this case, because of the **if** instruction (Line 7), the value of p is set to k and $\ell_i = \ell_{i+1} + 1$. Then, $\alpha = 1$ and so (**if** instruction of Line 10) $p_i = k - 1$. We will prove that $\lambda_k^L(T[i, d^*]) = \ell_{i+1}$ and $\pi(T[i, d^*]) = 1$. By the induction hypothesis, $\lambda_k^L(T[i+1, d^*]) = \ell_{i+1} - 1 \geq \lambda_i$ and $\pi(T[i+1, d^*]) = k$.

  We first prove that $\lambda_k^L(T[i, d^*]) \geq \lambda_k^L(T[i+1, d^*]) + 1 = \ell_{i+1}$. For purpose of contradiction, let us assume that $\lambda_k^L(T[i, d^*]) < \ell_{i+1}$ and let Φ be a k-strategy for $T[i, d^*]$ locating the target in at most $\ell_{i+1} - 1$ steps. First, if a node of $T[i]$ is probed during the first step of Φ, it means that at most $k - 1 < k - p_{i+1} = k$ nodes of $T[i+1, d^*]$ are probed during the first step of Φ, contradicting that $k - p_{i+1} = \pi(T[i+1, d^*])$ is the minimum number of nodes of $T[i+1, d^*]$ that must be probed during the first step of an optimal k-strategy for $T[i+1, d^*]$.

  Hence, neither Φ nor any k-strategy locating the target in $T[i, d^*]$ in at most $\ell_{i+1} - 1$ steps can probe some node of $T[i]$ during its first step. Below, we will build such a strategy Φ' (that probes some nodes of $T[i]$ during its first step) from Φ, which leads to a contradiction.

  Since Φ does not probe any node of $T[i]$ during its first step, then $\ell_{i+1} - 1 > \lambda_i$ (otherwise, a target hidden in $T[i]$ will not be located in at most $\ell_{i+1} - 1$ steps, by definition of $\lambda_i > 0$). Let $x > 1$ be the first step of Φ that probes a node of $T[i]$ if the target is in $T[i]$ (such a step exists since $T[i]$ is not a rooted path by definition of d, *i.e.*, since $\lambda_i > 0$). Then, $\ell_{i+1} - x \geq \lambda_i$ since, otherwise, a target hidden in $T[i]$ could not be located by Φ in at most $\ell_{i+1} - 1$ steps. If $\ell_{i+1} - x = \lambda_i$, then the x^{th} step of Φ must probe π_i nodes of $T[i]$. Otherwise, if $\ell_{i+1} - x > \lambda_i$, we may assume that the x^{th} step of Φ probes a single node of $T[i]$ (by Observation 4.8).

  Let $i+1 \leq j \leq d^*$ be such that the first step of Φ probes some nodes of $T[j]$. Because the subtrees have been sorted, $\lambda_j \leq \lambda_i < \ell_{i+1} - 1$ and we may assume that the first step of Φ probes one node in $T[j]$ (by Observation 4.8). Let us define the k-strategy Φ' as follows. The strategy Φ' follows Φ but, during its first step, it probes one node of $T[i]$ instead of probing some nodes of $T[j]$. If the target is located in $T[i]$, then Φ' applies an optimal strategy in $T[i]$ and locates the target in at most $\lambda_i < \ell_{i+1} - 1$ extra steps. Otherwise, Φ' continues to mimic the strategy Φ until its x^{th} step. If the target has been located in some subtree before the x^{th} step, then Φ' continues to act as Φ. Note that, during its step x, the strategy Φ probes at least one node of $T[i]$, and it probes at least π_i nodes of $T[i]$ if $\ell_{i+1} - x = \lambda_i$. Therefore, there are two cases to be considered.

  - If $\ell_{i+1} - x > \lambda_j$, then Φ' probes one node of $T[j]$ during step x. If the target is located in $T[j]$, then the next steps of Φ' follow an optimal strategy in $T[j]$ and will locate the target in at most λ_j extra steps. Otherwise, the next steps of Φ' follow the ones of Φ.
  - If $\ell_{i+1} - x = \lambda_j$, then it implies that $\ell_{i+1} - x = \lambda_i$ (since $\ell_{i+1} - x \geq \lambda_i \geq \lambda_j$) and that the step x of Φ was probing π_i nodes in $T[i]$. Note that, since the subtrees are ordered in non-increasing lexicographical order, $j > i$, and, $\lambda_j = \lambda_i$, then

$\pi_j \leq \pi_i$. The strategy Φ' replaces these π_i probes by probing $\pi_j \leq \pi_i$ nodes of $T[j]$. If the target is located in $T[j]$, then the next steps of Φ' follow an optimal strategy in $T[j]$ and will locate the target in at most $\lambda_j - 1$ extra steps. Otherwise, the next steps of Φ' follow the ones of Φ.

It is easy to show that Φ' is a k-strategy for $T[i, d^*]$ locating the target in at most $\ell_{i+1} - 1$ steps, and probing some node of $T[i]$ during its first step, a contradiction.

We now prove that $\lambda_k^L(T[i, d^*]) = \ell_{i+1}$ and that $\pi(T[i, d^*]) = 1$. It is sufficient to design a strategy Φ for $T[i, d^*]$ locating the target in at most ℓ_{i+1} steps. By Observation 4.8, there is a strategy Φ' for $T[i]$ for locating the target in at most $\lambda_i + 1$ steps and probes a single node during the first step. Also, let Φ'' be an optimal strategy for $T[i+1, d^*]$. The first step of Φ consists of probing one node of $T[i]$. If the target is in $T[i]$, then the strategy continues with Φ' (in at most $\lambda_i \leq \ell_{i+1} - 1$ extra steps), otherwise, the strategy continues with Φ'' (in at most $\lambda_k^L(T[i+1, d^*]) = \ell_{i+1} - 1$ extra steps). From this, we deduce that $\pi(T[i, d^*]) \leq 1$ and, by definition of π, we get that $\pi(T[i, d^*]) = 1$.

- Case $p_{i+1} > 0$, $\ell_{i+1} > \lambda_i + 1$.

  In this case, the condition of the **if** instruction (Line 7) is not satisfied, $\alpha = 1$, and so the condition of the **if** instruction (Line 10) is satisfied. Hence, $\ell_i = \ell_{i+1}$ and $p_i = p_{i+1} - 1$. We will prove that $\lambda_k^L(T[i, d^*]) = \ell_{i+1} - 1$ and $\pi(T[i, d^*]) = k - p_{i+1} + 1$. By the induction hypothesis, we have $\lambda_k^L(T[i+1, d^*]) = \ell_{i+1} - 1 > \lambda_i$ and $\pi(T[i+1, d^*]) = k - p_{i+1}$. By Observation 4.7, $\lambda_k^L(T[i, d^*]) \geq \lambda_k^L(T[i+1, d^*]) = \ell_{i+1} - 1$ also.

  To prove that $\lambda_k^L(T[i, d^*]) \leq \lambda_k^L(T[i+1, d^*]) = \ell_{i+1} - 1$, it is sufficient to describe a strategy Φ for $\lambda_k^L(T[i, d^*])$ with a total of at most $\ell_{i+1} - 1$ steps. By Observation 4.8, there is a strategy Φ' for $T[i]$ for locating the target in at most $\lambda_i + 1$ steps that probes a single node during the first step. Let Φ'' be an optimal strategy for $T[i+1, d^*]$ probing at most $\pi(T[i+1, d^*]) = k - p_{i+1} < k$ nodes during the first step. The first step of Φ consists of probing one node in $T[i]$ (as Φ') and $\pi(T[i+1, d^*]) = k - p_{i+1}$ nodes of $T[i+1, d^*]$ (as Φ''). By assumption, $0 < p_{i+1}$, so $1 + \pi(T[i+1, d^*]) \leq k$ and at most k nodes are probed. By Claim 4.4, this first step allows to decide if the target is in $T[i]$ or not (in which case it is in $T[i+1, d^*]$). If the target is in $T[i]$, then Φ continues by following the strategy Φ' in $T[i]$ which will locate the target in at most $\lambda_i < \ell_{i+1} - 1$ extra steps. Otherwise (the target is in $T[i+1, d^*]$), Φ continues by following the optimal strategy Φ'' for $T[i+1, d^*]$ which will locate the target in at most $\lambda_k^L(T[i+1, d^*]) - 1 = \ell_{i+1} - 2$ extra steps. In all cases, Φ locates the target in at most $\ell_{i+1} - 1$ steps.

  Let us prove that $\pi(T[i, d^*]) = k - p_{i+1} + 1$. For purpose of contradiction, let us assume that there is a strategy Φ for locating the target in $T[i, d^*]$ in at most $\ell_{i+1} - 1$ steps that probes strictly less than $k - p_{i+1} + 1$ nodes during the first step. We will show that we can construct a strategy Φ' in $T[i+1, d^*]$ for locating the target in at most $\ell_{i+1} - 1$ steps and probing at most $k - p_{i+1} - 1$ nodes during the first step, a contradiction. If the first step of Φ probes at least one node of $T[i]$, then it probes at most $k - p_{i+1} - 1$ nodes of $T[i+1, d^*]$ contradicting the fact that $\lambda_k^L(T[i+1, d^*]) = \ell_{i+1} - 1$ and $\pi(T[i+1, d^*]) = k - p_{i+1}$. Hence, we may assume that the first step of Φ probes at least $k - p_{i+1}$ nodes of $T[i+1, d^*]$ and no nodes in $T[i]$.

Let $x > 1$ be the minimum integer such that at least one node of $T[i]$ is probed during the x^{th} step of Φ. After step x, at most $\ell_{i+1} - x - 1$ steps remain and so $\ell_{i+1} - x - 1 \geq \lambda_i - 1$. Let $j \in [\![i+1, d^*]\!]$ be such that at least one node of $T[j]$ is probed during the first step of Φ. Note that $j > i$ and, because the subtrees are ordered in non-increasing lexicographical order, either $\lambda_j < \lambda_i$ or ($\lambda_j = \lambda_i$ and $\pi_j \leq \pi_i$).

Let us consider the following strategy Φ' for $T[i+1, d^*]$. The first $x-1$ steps of the strategy Φ' follow the ones of Φ but do not probe any node of $T[j]$. That is, for every $j' \in [\![i+1, d^*]\!] \setminus \{j\}$ and for every $x' < x$, the step x' of Φ' probes the same nodes of $T[j']$ as the step x' of Φ. In particular, the first step of Φ' probes at most $k - p_{i+1} - 1$ nodes. If the target has been located in a subtree different from $T[j]$ during the first $x-1$ steps, then Φ' continues as Φ (but without probing the nodes of $T[i]$ since Φ' is a strategy for $T[i+1, d^*]$). Otherwise, the x^{th} step of Φ' proceeds as follows. For every $j' \in [\![i+1, d^*]\!] \setminus \{j\}$, the step x of Φ' probes the same nodes of $T[j']$ as the step x of Φ. Again, the strategy Φ' does not probe any node of $T[i]$. Note that, during its step x, the strategy Φ probes at least one node of $T[i]$, and it probes at least π_i nodes of $T[i]$ if $\ell_{i+1} - x = \lambda_i$. Therefore, there are two cases to be considered.

- If $\ell_{i+1} - x > \lambda_j$, then Φ' probes one node of $T[j]$ during step x. If the target is located in $T[j]$ then the next steps of Φ' follow an optimal strategy in $T[j]$ and will locate the target in at most λ_j extra steps. Otherwise, the next steps of Φ' follow the ones of Φ.
- If $\ell_{i+1} - x = \lambda_j$, then it implies that $\ell_{i+1} - x = \lambda_i$ (since $\ell_{i+1} - x \geq \lambda_i \geq \lambda_j$) and that the step x of Φ was probing π_i nodes in $T[i]$. Note that, since the subtrees are ordered in non-increasing lexicographical order, $j > i$, and, $\lambda_j = \lambda_i$, then $\pi_j \leq \pi_i$. The strategy Φ' replaces these π_i probes by probing $\pi_j \leq \pi_i$ nodes of $T[j]$. If the target is located in $T[j]$ then the next steps of Φ' follow an optimal strategy in $T[j]$ and will locate the target in at most $\lambda_j - 1$ extra steps. Otherwise, the next steps of Φ' follow the ones of Φ.

Overall, Φ' is a strategy for locating the target in $T[i+1, d^*]$ in at most $\ell_{i+1} - 1$ steps that probes at most $k - p_{i+1} - 1$ nodes during the first step. This contradicts the fact that $\pi(T[i+1, d^*]) = k - p_{i+1}$. Hence, $\pi(T[i, d^*]) = k - p_{i+1} + 1$.

- Case $\ell_{i+1} < \lambda_i + 1$.

  In this case, because of the **if** instruction (Line 7), the value of p is set to k and $\ell_i = \lambda_i + 1$. Then, $\alpha = \pi_i$ and so (**if** instruction on Line 10) $p_i = k - \pi_i$. We will prove that $\lambda_k^L(T[i, d^*]) = \lambda_i$ and $\pi(T[i, d^*]) = \pi_i$. By the induction hypothesis, $\lambda_k^L(T[i+1, d^*]) = \ell_{i+1} - 1 < \lambda_i$ and $\pi(T[i+1, d^*]) = k - p_{i+1}$. Also, by Observation 4.7, $\lambda_k^L(T[i, d^*]) \geq \lambda_k^L(T[i]) = \lambda_i$.

  To prove that $\lambda_k^L(T[i, d^*]) \leq \lambda_i$, it is sufficient to describe a strategy Φ for $\lambda_k^L(T[i, d^*])$ with a total of at most λ_i steps. Let Φ' be an optimal strategy for $T[i]$ probing at most π_i nodes during the first step. Let Φ'' be an optimal strategy for $T[i+1, d^*]$ for locating the target in at most $\ell_{i+1} - 1 < \lambda_i$ steps and probing at most $\pi(T[i+1, d^*]) = k - p_{i+1}$ nodes during the first step. The first step of Φ probes π_i nodes of $T[i]$ (as Φ'). By Claim 4.4, this first step allows to decide if the target is in $T[i]$ or not. If it is in $T[i]$, then Φ follows Φ'. Otherwise, Φ executes Φ'' in $T[i+1, d^*]$.

  To conclude, let us prove that $\pi(T[i, d^*]) = \pi_i$. The previous strategy Φ shows that $\pi(T[i, d^*]) \leq \pi_i$. Since $\lambda_k^L(T[i, d^*]) = \lambda_i$, any strategy for $T[i, d^*]$ must probe at least π_i nodes of $T[i]$ during the first step by definition of π_i. This concludes the proof. □

With Lemma 4.9 in hand, we are now ready to prove that Algorithms 1 and 2 are correct. We also prove that their running time is polynomial. More precisely, we prove in Theorem 4.11 below that $\mathcal{A}_1(k,(T,r))$ computes $(\lambda_k^L(T),\pi(T))$ and a corresponding k-strategy in time $O(n\log n)$, where n is the number of nodes. To do that, Theorem 4.10 proves the correctness and the linear (in the number of children of r) time complexity of $\mathcal{A}_2(k,(T,r),(\Lambda,\Pi))$.

Theorem 4.10. *Let $k\geq 1$ and $(T,r)\in\mathcal{T}$ be a tree rooted in r with children $v_1,\ldots,v_{d^*}$, such that the tuples $(\Lambda,\Pi)=(\lambda_i,\pi_i)_{1\leq i\leq d^*}$ are sorted in non-increasing lexicographical ordering. Then, $\mathcal{A}_2(k,(T,r),(\Lambda,\Pi))$ returns $(\lambda_k^L(T),\pi(T))$ and a corresponding strategy. Furthermore, the time complexity of $\mathcal{A}_2$ is $O(d^*)$ (independent of k).*

Proof. The time complexity is obvious and the correctness follows from Lemma 4.9 for $i=1$. The fact that the strategy is also returned is not explicitly described in Algorithm 2 but directly follows from the proof of Lemma 4.9. □

Theorem 4.11. *Let $k\geq 1$ and let $(T,r)\in\mathcal{T}$ be an n-node rooted tree. Then, $\mathcal{A}_1(k,(T,r))$ returns $(\lambda_k^L(T),\pi(T))$ and a corresponding strategy. Furthermore, the time complexity of $\mathcal{A}_1$ is $O(n\log n)$ (independent of k).*

Proof. The correctness is simply proved by induction on the depth of the tree and by Theorem 4.10. For the time complexity, at every recursive call on a subtree T_v rooted at v (with d_v children), the additional number of operations is $O(d_v\log d_v)$ (sorting) plus $O(d_v)$ (Algorithm $\mathcal{A}_2$, by Theorem 4.10). Since, in any n-node tree T, $\sum_{v\in V(T)} d_v = 2(n-1)$, this gives a total complexity of $O(\sum_{v\in V(T)} d_v\log d_v) = O(n\log n)$. Again, the strategy is not explicit in our presentation but can be easily computed. □

4.3 A polynomial-time $(+1)$-approximation

From Algorithm $\mathcal{A}_1(k,(T,r))$, it is easy to get an efficient approximation algorithm for solving Localization in trees when k and ℓ are part of the input, and a polynomial time algorithm when k is fixed.

Theorem 4.12. *There exists an algorithm with running time $O(n\log n)$ that, given any integer $k\geq 1$ and n-node tree T, computes a k-strategy for locating a target in T in at most $\lambda_k(T)+1$ steps.*

Proof. The strategy proceeds as follows. The first step probes any arbitrary node r of T. Let d be the distance between r and the target, $L\subseteq V(T)$ be the set of nodes at distance exactly d from r, and T^d be the subtree induced by r and every node on a path between r and the nodes in L. Note that $(T^d,r)\in\mathcal{T}$ and that the target is occupying a leaf of T^d. Hence, the target can be located by applying $\mathcal{A}_1(k,(T^d,r))$. By Theorem 4.11, this will locate the target in at most $1+\max_d \lambda_k^L(T^d)\leq 1+\lambda_k(T)$ steps. □

Theorem 4.13. *There exists an algorithm with running time $O(n^{k+1}\log n)$ that, given any integer $k\geq 1$ and n-node tree T, computes an optimal k-strategy for locating a target in T in at most $\lambda_k(T)$ steps.*

Proof. The proof is similar to the one of Theorem 4.12, but instead of probing a single node during the first step, we enumerate all of the $O(n^k)$ possibilities for the first step, and, for each of them, we then apply Algorithm $\mathcal{A}_1$. For one of these instances, the target will be located within $\lambda_k(T)$ steps. □

To conclude this section, it is important to mention that both Theorems 4.12 and 4.13 also hold in the case of edge-weighted trees. Indeed, distances are only used in Claim 4.4 which clearly holds for edge-weighted trees.

5 Conclusion and further work

In this work, we have studied the computational complexity of the Localization problem. We have established the importance of its two main parameters, namely the number k of vertices that can be probed each step, and the number ℓ of steps within which the target must be located. Namely, fixing any of these two parameters makes the problem NP-complete. This remains true for the Relative-Localization problem as well.

We have then focused on the case of trees. For that case, we have proved that Localization remains NP-complete. However, the only source of hardness is due to the first probing step as, as soon as the second step begins, an optimal way to play can be computed in polynomial time. As a consequence, the problem, though hard, can be approximated efficiently.

Our results in trees leave the open question of whether determining $\lambda_k(T)$ is FPT (in k) in the class of trees T. Also, we do not know the complexity of determining whether $\kappa_\ell(T) \leq k$ for a tree T. An interesting line of research could be to study all those problems in other graph classes, such as interval or planar graphs.

The problem of determining $\lambda_k^{rel}(T)$ for a tree T seems to be much more intricate even for simple topologies. A first step towards a better understanding of it would be to fully understand the centroidal dimension of paths (*i.e.*, to determine $\kappa_1^{rel}(P)$ for every path P), which has been initiated in [9]. A more challenging direction would then be to consider the case of all trees.

References

[1] Rémy Belmonte, Fedor V. Fomin, Petr A. Golovach, and M. S. Ramanujan. Metric dimension of bounded tree-length graphs. *SIAM J. Discrete Math.*, 31(2):1217–1243, 2017.

[2] Y. Ben-Haim, S. Gravier, A. Lobstein, and J. Moncel. Adaptive identification in graphs. *Journal of Combinatorial Theory, Series A*, 115(7):1114–1126, 2008.

[3] Julien Bensmail, Dorian Mazauric, Fionn Mc Inerney, Nicolas Nisse, and Stéphane Pérennes. Sequential metric dimension. In *Proceedings of the 16th Workshop on Approximation and Online Algorithms, WAOA 2018*, volume 11312 of *Lecture Notes in Computer Science*, pages 36–50. Springer, 2018.

[4] Bartlomiej Bosek, Przemyslaw Gordinowicz, Jaroslaw Grytczuk, Nicolas Nisse, Joanna Sokól, and Malgorzata Sleszynska-Nowak. Centroidal localization game. *Electronic Journal of Combinatorics*, 25(4):P4.62, 2018.

[5] Bartlomiej Bosek, Przemyslaw Gordinowicz, Jaroslaw Grytczuk, Nicolas Nisse, Joanna Sokól, and Malgorzata Sleszynska-Nowak. Localization game on geometric and planar graphs. *Discrete Applied Mathematics*, 251:30–39, 2018.

[6] A. Brandt, J. Diemunsch, C. Erbes, J. LeGrand, and C. Moffatt. A robber locating strategy for trees. *Discrete Applied Mathematics*, 232:99 – 106, 2017.

[7] James M. Carraher, Ilkyoo Choi, Michelle Delcourt, Lawrence H. Erickson, and Douglas B. West. Locating a robber on a graph via distance queries. *Theor. Comput. Sci.*, 463:54–61, 2012.

[8] Josep Díaz, Olli Pottonen, Maria J. Serna, and Erik Jan van Leeuwen. Complexity of metric dimension on planar graphs. *J. Comput. Syst. Sci.*, 83(1):132–158, 2017.

[9] Florent Foucaud, Ralf Klasing, and Peter J. Slater. Centroidal bases in graphs. *Networks*, 64(2):96–108, 2014.

[10] Florent Foucaud, George B. Mertzios, Reza Naserasr, Aline Parreau, and Petru Valicov. Identification, location-domination and metric dimension on interval and permutation graphs. i. bounds. *Theor. Comput. Sci.*, 668:43–58, 2017.

[11] Florent Foucaud, George B. Mertzios, Reza Naserasr, Aline Parreau, and Petru Valicov. Identification, location-domination and metric dimension on interval and permutation graphs. II. algorithms and complexity. *Algorithmica*, 78(3):914–944, 2017.

[12] M. R. Garey and D. S. Johnson. *Computers and Intractability - A guide to NP-completeness.* W.H. Freeman and Company, 1979.

[13] Frank Harary and Robert A. Melter. On the metric dimension of a graph. *Ars Combinatoria*, 2:191–195, 1976.

[14] Sepp Hartung and André Nichterlein. On the parameterized and approximation hardness of metric dimension. In *Proceedings of the 28th Conference on Computational Complexity, CCC*, pages 266–276. IEEE Computer Society, 2013.

[15] John Haslegrave, Richard A. B. Johnson, and Sebastian Koch. Locating a robber with multiple probes. *Discrete Mathematics*, 341(1):184–193, 2018.

[16] Mark G. Karpovsky, Krishnendu Chakrabarty, and Lev B. Levitin. On a new class of codes for identifying vertices in graphs. *IEEE Trans. Information Theory*, 44(2):599–611, 1998.

[17] Suzanne M. Seager. Locating a robber on a graph. *Discrete Mathematics*, 312(22):3265–3269, 2012.

[18] Suzanne M. Seager. A sequential locating game on graphs. *Ars Comb.*, 110:45–54, 2013.

[19] Suzanne M. Seager. Locating a backtracking robber on a tree. *Theor. Comput. Sci.*, 539:28–37, 2014.

[20] Peter J. Slater. Leaves of trees. pages 549–559. Congressus Numerantium, No. XIV, 1975.

[21] Peter J. Slater. Domination and location in acyclic graphs. *Networks*, 17(1):55–64, 1987.