Quasilinear-time eccentricities computation, and more, on median graphs

Pierre Bergé, Guillaume Ducoffe, Michel Habib

Quasilinear-time eccentricities computation, and more, on median graphs

Pierre Bergé[1], Guillaume Ducoffe[2,3], and Michel Habib[4]

[1]Université Clermont-Auvergne, CNRS, Mines de Saint-Etienne, Clermont-Auvergne-INP, LIMOS, 63000 Clermont-Ferrand, France
[2]National Institute for Research and Development in Informatics, Romania
[3]University of Bucharest, Romania
[4]IRIF, CNRS & Université Paris Cité, France

Abstract

Computing the diameter, and more generally, all eccentricities of an undirected graph is an important problem in algorithmic graph theory and the challenge is to identify graph classes for which their computation can be achieved in subquadratic time. Using a new recursive scheme based on the structural properties of median graphs, we provide a quasilinear-time algorithm to determine all eccentricities for this well-known family of graphs. Our recursive technique manages specifically balanced and unbalanced parts of the Θ-class decomposition of median graphs. The exact running time of our algorithm is $O(n \log^4 n)$. This outcome not only answers a question asked by Bénéteau *et al.* (2020) but also greatly improves a recent result which presents a combinatorial algorithm running in time $O(n^{1.6408} \log^{O(1)} n)$ for the same problem.

Furthermore we also propose a distance oracle for median graphs with both polylogarithmic size and query time. Speaking formally, we provide a combinatorial algorithm which computes for any median graph G, in quasilinear time $O(n \log^4(n))$, vertex-labels of size $O(\log^3(n))$ such that any distance of G can be retrieved in time $O(\log^4(n))$ thanks to these labels.

1 Presentation of the contributions: ideas and impact

A wide literature in graph theory, and more specifically in metric graph theory, is dedicated to median graphs. Given two vertices $u, v \in V(G)$, the interval $I(u, v)$ is the set containing all vertices x metrically between u and v, *i.e.* $x \in I(u, v)$ if $d(u, v) = d(u, x) + d(x, v)$. Formally, median graphs are the graphs G such that for any triplet of distinct vertices $u, v, w \in V(G)$, the intersection $I(u, v) \cap I(v, w) \cap I(w, u)$ is a singleton. From the applications point of view, median graphs are crucial in the study of phylogenetic networks [8, 36]. They are usually mentioned as "median networks" in this area of research. They are, in particular, widely considered in the visualization of sequence variations in human mitochondrial DNA [7, 9, 11, 41]. From a more theoretical point of view, median graphs are in bijection with numerous and diverse notions from discrete mathematics. For example, they represent solutions of 2-SAT

formulae [33, 38]. In geometric group theory, median graphs are exactly the 1-skeletons of CAT(0) cube complexes [4, 21]. In abstract models of concurrency, median graphs are in bijection with event structures [12, 37]. Finally, median graphs form a natural subclass of *partial cubes*, *i.e.* isometric subgraphs of hypercubes [34].

From a graph theory point of view now, median graphs admit many characterizations and structural properties. First, median graphs are bipartite (hence, triangle-free) and do not admit any induced $K_{2,3}$. They are exactly the retracts of hypercubes [2], and the Cartesian product of two median graphs is also median [4]. Median graphs are sparse: the number m of edges of any median graph G satisfies $m \leq n \log_2 n$, where $n = |V(G)|$. Among the most famous subclasses of median graphs, we can cite: trees, hypercubes, grids (of any dimension), and cogwheels. A very practical characterization of median graphs come from the notion of Θ-classes which provide us with natural edge separators satisfying convexity properties. This concept, used in most of the algorithmic literature cited below, will be a key instrument of our contributions in this article.

Recently, several efficient algorithms, dedicated to solving distance problems on median graphs, have been proposed [14, 15, 16, 20, 23, 26]. An important outcome is the algorithm proposed by Bénéteau *et al.* [14] which computes both the median set and the Wiener index of median graphs G in linear time $O(m)$, where $m = |E(G)|$. A second one [15] computes all eccentricities of median graphs in subquadratic time $\tilde{O}(n^{1.6408})$, where the $\tilde{O}$ notation neglects poly-logarithmic factors. Constant 1.6408 comes from a slight improvement on a first version of the algorithm, which achieved $\tilde{O}(n^{\frac{5}{3}})$.

Other contributions focused on subclasses of median graphs too. There is a linear-time algorithm [16] which determines the diameter of median graphs for which the dimension of the largest induced hypercube is bounded. Furthermore, Chepoi *et al.* [23] designed distance and routing labeling schemes of $O(\log^3 n)$ bits for cube-free median graphs.

Our contributions. The main outcome of this paper consists in the proposal of a combinatorial algorithm which computes all eccentricites of a median graph in quasilinear time. Before stating formally this contribution, let us recall the problem we treat here. The distance $d(u, v)$ between two vertices $u, v \in V(G)$ is the length of a shortest (u, v)-path. Given a vertex $u \in V(G)$, its *eccentricity* $\mathsf{ecc}(u \mid G)$ is the maximum distance from u to any other vertex of G. The *diameter* and *radius*, which are certainly the most studied metric parameters on graphs, correspond respectively to the maximum/minimum eccentricity of the graph.

Eccentricities
Input: A median graph $G = (V, E)$.
Output: All labels $\mathsf{ecc}(u \mid G) = \max\{d(u, v) : v \in V\}$ for each vertex $u \in V$.

In this article, we focus in fact on a more general problem, which is a weighted version of the Eccentricities problem. The input graph is a vertex-weighted median graph and the objective is to determine, for each vertex u, its *weighted eccentricity* $\mathsf{ecc}(u \mid (G, \omega))$, which is the maximum value $d(u, v) + \omega(v)$. In other words, the weight of the arrival vertex is added to the standard distance. Obviously, it generalizes Eccentricities since fixing weights 0 to each vertex is equivalent to the classical problem.

Weighted Eccentricities
Input: A weighted median graph $G = (V, E, \omega)$, with $\omega : V \to \mathbb{N}$.
Output: All labels $\mathsf{ecc}(u \mid (G, \omega)) = \max\{d(u, v) + \omega(v) : v \in V\}$ for each $u \in V$.

The main contribution of this article is thus stated below. Observe that it greatly enhances the literature as, for now, the best running time for computing all eccentricies of a median graph was in $\tilde{O}(n^{1.6408})$, from [15]. Moreover, we handle a more general problem.

Theorem 1. *There exists a combinatorial algorithm which computes all weighted eccentricities of a weighted median graph (G, ω) in quasilinear time $O(n \log^4(n))$.*

This result is described in Sections 3 and 4. The idea of our algorithm is in fact relatively simple: Θ-classes, which are presented in the preliminary Section 2, are separators of the input graph G and admit several convexity properties. Hence, inspired by well-known divide-and-conquer methods on graphs [31], we make a tradeoff on the balance of these separators: either the sizes of the two sides generated by the separator are comparable, or not. Thus, we distinguish two cases.

When the median graph G admits a balanced Θ-class (this notion is defined in Section 3.1), we retrieve all weighted eccentricities by recursively focusing on each side of the separator, with an extra procedure that runs in linear time. As the Θ-class is balanced, there is a non-negligible decrease of the size of each side compared to $n = |V(G)|$. After obtaining the weighted eccentricities of each side recursively, our extra procedure consists in retrieving the weighted eccentricities of the whole graph by looking for large distances between vertices of different sides. This can be achieved thanks to the *gatedness* of each side: this property of Θ-classes will be recalled in Section 2.1.

However, when no Θ-class of the input graph is balanced, the previous technique is not efficient anymore since for any Θ-class, the size of its sides does not decrease enough. We provide a list of characterizations of median graphs without balanced Θ-classes. In particular, we observe that such graphs admit a unique median vertex v_0. Then, we prove how a BFS starting from v_0 together with a reasonable number of recursive calls on convex subgraphs help us in finding all weighted eccentricities of (G, ω). This second part of our algorithm is more technical than the first one.

As a second result, we propose a *distance oracle* (DO) for median graphs. The objective beyond our DO is to locally store some information so that one can retrieve the distance between any pair of vertices by inspecting the labels in a very short time. Concretely, we propose an algorithm which assigns a label to each vertex of the graph, and then we show how these labels can be used to compute fast any value $d(u, v)$. Observe however that our labeling is not a *distance labeling scheme*, since we might need to look at more than two vertex labels in order to compute some distance $d(u, v)$. Indeed, the information given by the labels of u and v might not be sufficient to obtain $d(u, v)$, extra labels must be taken into account. Our result is described in Section 5.

Theorem 2. *There exists a combinatorial algorithm which computes in quasilinear time $O(n \log^4(n))$, for any median graph G, vertex-labels $(\Lambda_G(u))_{u \in V(G)}$ of size $O(\log^3(n))$ such that the distance $d(x, y)$ between a pair x, y of vertices can be retrieved in time $O(\log^4(n))$ thanks to the labels.*

The proof re-uses the splitting between the balanced and unbalanced cases. When a Θ-class is balanced, we apply similar arguments to those used for the computation of eccentricities, with the difference that the necessary information is put into vertex labels. Concretely, any vertex u is labeled with the identity of the balanced Θ-class considered, the side of u, the

closest-to-u vertex on the other side (its *gate*), the distance to its gate, and finally the label of u in the graph induced by its side regarding the balanced Θ-class. This whole package allows us to retrieve any distance between two vertices in poly-logarithmic time.

The unbalanced case requires more effort. Our idea consists in partitioning all vertices of the graph in function of their "direction" regarding the central vertex v_0. To do so, we launch a BFS from v_0 and take note of some information, that will be added to the label of any vertex $u \neq v_0$: the distance from v_0 to u, the Θ-classes traversed to go from v_0 to u, etc. In addition, some gates of u through certain small convex sets are also computed. With this information, we show again how to retrieve any distance in poly-logarithmic time.

Perspectives. First, we believe that the techniques proposed in this article offer not only tools for different problems on median graphs but also for more general classes of graphs. Observe that larger families of graphs are still impacted by the notion of Θ-class, the main difference consisting in weaker convex characterizations: almost median graphs [17], pseudo-median graphs [10, 39] or partial cubes [40]. In our work, we exploit several times the fact that the *boundary* of each Θ-class is convex, which is a property specific to median graphs and not to these superclasses. Therefore, one should be able to get rid of this argument in order to handle larger families of graphs. However, a tradeoff on the balance of Θ-class stays, in our opinion, a promising starting point for tackling them.

Coming back to median graphs, one can hope producing efficient algorithms by exploiting again this tradeoff technique. A future direction of research could be trying to design algorithms which improve the naive general method for computing other metric parameters, such as the *hyperbolicity* [27], the *betweenness* and *reach centralities* [1],... The techniques proposed in our paper can be useful tools for such problems. Eventually, we mention a problem which was not studied yet on median graphs to the best of our knowledge: Weighted Center [13]. Given a median graph G with vertex weights $\omega : V \to \mathbb{N}$, the objective is to determine the *weighted center* of (G, ω), *i.e.* the vertex u which minimizes $\max_{v \in V(G)} \omega(v)d(u, v)$. Observe that, with Theorem 1, we can deduce, from the weighted eccentricities, some kind of weighted center where weights stand as an additive term and not multiplicative. For this reason, we believe that the techniques we proposed can be fruitful for solving Weighted Center in quasilinear time. Note that Weighted Center admits exact quasilinear-time algorithms for trees and cactii [13], hence targeting median graphs is a natural challenge.

2 Preliminaries

We begin with a reminder of some notions of graphs, and more particularly on median graphs. We emphasize on a very important tool in this area: Θ-classes, which are equivalences classes over the edge set of median graphs, and especially the orthogonality between these Θ-classes. From the mathematical point of view, notation log refers to the natural logarithm, *i.e.* $\log n = \log_e n$. When another base is considered, we mention it as a subscript, *e.g.* $\log_2$.

2.1 Definitions and properties from metric graph theory

All graphs $G = (V, E)$ considered in this paper are undirected, simple (loopless and without multiple edges), finite and connected. To avoid confusions when several graphs are considered, we denote by $V(G)$ (resp. $E(G)$) the vertex set (resp. edge set) of G. Usually, we use n to denote the size of the vertex set of G, *i.e.* $|V(G)|$, while m denotes the size of the edge set:

$m = |E(G)|$. To improve readibility, edges $(u, v) \in E$ are sometimes denoted by uv. Let $N(u)$ be the *open neighborhood* of $u \in V$, *i.e.* the set of vertices adjacent to u in G. We extend it naturally: for any set $A \subseteq V$, the neighborhood $N(A)$ of A is the set of vertices outside A adjacent to some $u \in A$.

A subgraph G' of G is a graph $G' = (V', E')$, where $V' \subseteq V$ and $E' \subseteq E \cap (V' \times V')$. For any $U \subseteq V$, let $E[U]$ be the set of edges of G with two endpoints in U. We denote by $G[U]$ the subgraph of G induced by U: $G[U] = (U, E[U])$.

Given two vertices $u, v \in V$, let $d(u, v)$ be the *distance* between u and v, *i.e.* the length of a shortest (u, v)-path. When the context is not clear enough, we might notice, as a subscript, in which graph these distance are considered, *e.g.* $d_G(u, v)$. As a weighted version is defined in the remainder, we may refer to this notion as the *unweighted distance*. The *eccentricity* $\mathsf{ecc}(u \mid G)$ of a vertex $u \in V$ in graph G is the length of a longest shortest path starting from u. Put formally, $\mathsf{ecc}(u \mid G)$ is the maximum value $d(u, v)$ for all $v \in V$: $\mathsf{ecc}(u \mid G) = \max_{v \in V} d(u, v)$. The diameter of graph G is the maximum distance between two of its vertices: $\mathsf{diam}(G) = \max_{u \in V} \mathsf{ecc}(u \mid G)$.

We denote by $I(u, v)$ the *interval* of pair $u, v \in V$. It contains exactly the vertices which are metrically between u and v: $I(u, v) = \{x \in V : d(u, x) + d(x, v) = d(u, v)\}$. The vertices of $I(u, v)$ are lying on at least one shortest (u, v)-path.

Definition 1 (Convex and gated sets)**.** *We say that a set $H \subseteq V$ (or equivalently the induced subgraph $G[H]$) is* convex *if $I(u, v) \subseteq H$ for any pair $u, v \in H$. Moreover, we say that H is* gated *if any vertex $v \notin H$ admits a H-*gate *$g_H(v) \in H$,* i.e. *a unique vertex that belongs to all intervals $I(v, x)$, $x \in H$. For any $x \in H$, we have $d(v, g_H(v)) + d(g_H(v), x) = d(v, x)$.*

Observe that, if $v \in H$, then it admits a natural H-gate: itself, since v belongs not only to H but also to all intervals $I(v, x)$, $x \in H$. Gated sets are convex by definition. Indeed, by contradiction, for a gated set H, if a shortest path from $x \in H$ to $y \in H$ was containing a section outside H, it would imply the existence of two H-gates for the vertices of this section. Conversely, convex sets are not necessarily gated in general (*e.g.* any pair of vertices in the 3-clique K_3). But, we will see in the remainder that, on median graphs, convexity and gatedness are equivalent notions.

A well-known property is that the intersection of two gated sets is itself gated.

Lemma 1 (Intersection of gated subgraphs [4])**.** *Given two gated sets H_1, H_2 of a graph G, the set $H_1 \cap H_2$ is gated.*

We naturally focus on the set of vertices which admit a given vertex as a gate.

Definition 2 (Fibers [4])**.** *Given a gated set H of G and a vertex $x \in H$, the fiber $F_H[x]$ is the set of vertices which admit $x \in H$ as a gate for H.*

As each vertex in H is its own gate, the fibers $F_H[x]$, $x \in H$, partition $V(G)$. A fiber is thus a set of vertices which all admit the same gate for a given gated H. In fact, the H-gate of some $v \notin H$ is necessarily the vertex of H minimizing the distance to v. Observe that, except x itself, $F_H[x]$ contains vertices outside H: $F_H[x] \setminus \{x\} \subseteq V(G) \setminus H$. In the following sections, we often manipulate this set $F_H[x] \setminus \{x\}$ that we call the *open fiber*.

Definition 3 (Open fibers)**.** *Given a gated set H of G and a vertex $x \in H$, we define the* open fiber *as $F_H(x) = F_H[x] \setminus \{x\}$.*

There is a natural linear-time algorithm (in fact, a very slight variation of BFS) for computing the fibers (and open fibers) of a given gated set H. To keep this paper as self-contained as possible, we recall a sketch of this algorithm, proposed in [23]. The classical BFS uses a queue to store the visited vertices of the graph. Here, we initialize this queue by putting into it all the vertices of H (instead of a single starting vertex). Each vertex $v \in V$ is labeled by three variables: $d(v)$ (distance to the gate), $f(v)$ (parent of v in the BFS tree) and $\texttt{fib}(v)$ (H-gate of v). All vertices $x \in H$ are initialized with $d(x) = 0$, $f(x) = \texttt{NULL}$, and $\texttt{fib}(x) = x$. Then, we proceed the search as follows. Once a vertex v is at the head of the queue, all not yet discovered neighbors w of v are inserted into the queue, and we fix $d(w) = d(v) + 1$, $f(w) = v$, $\texttt{fib}(w) = \texttt{fib}(v)$. At the end of the execution, each vertex v is labeled by not only its gate $\texttt{fib}(v) = g_H(v)$ in H but also the (unweighted) distance towards its gate $d(v)$. For the correctness of this BFS traversal, see Lemma 16 and Corollary 6 from [22], the open access version of [23].

Lemma 2 ([22, 23])**.** *For any gated set H of a graph G, one can compute, in linear time $O(m) = O(n \log n)$, all fibers $F_H[x]$, with $x \in H$, but also all distances from each $v \in V(G)$ to its H-gate.*

Let us denote by $(G, \omega) = ((V, E), \omega)$ the *weighted graph* G when it is equipped with an integer non-negative weight function on its vertices $\omega : V \to \mathbb{N}$. In brief, in this paper, expression *weighted graph* refers to vertex-weighted graphs. Observe that the definitions of the structural notions we introduced above are independent from any weight consideration. Notions of interval, gate, convex and gated sets, and fiber only depend on unweighted distances.

Now, considering a weighted graph (G, ω), the distance is generalized to a "*weighted distance*" which is not symmetrical anymore: $d_\omega(u, v) = d(u, v) + \omega(v)$. In particular, $d_\omega(u, u) = \omega(u)$. When there is some ambiguity on the graph considered, we may write the weighted distance as $d_{(G,\omega)}(u, v)$. The *weighted eccentricity* of a vertex $u \in V$ is the maximum weighted distance from this vertex, *i.e.* $\mathsf{ecc}\,(u \mid (G, \omega)) = \max_{v \in V} d_\omega(u, v)$. In future sections, this notion of weighted distances/eccentricities will allow us to describe recursive algorithms on median graphs based on gated sets.

2.2 Median graphs and Θ-classes

From now on, we focus on the family of graphs which is studied in this article: median graphs.

Definition 4 (Median graph)**.** *A graph is median if, for any triplet (x, y, z) of distinct vertices, the set $I(x, y) \cap I(y, z) \cap I(z, x)$ contains exactly one vertex $m(x, y, z)$ called the* median *of (x, y, z).*

Trees, hypercubes, grids and squaregraphs [6] are median graphs. Median graphs are bipartite and do not contain any induced $K_{2,3}$ [4, 29, 34]. The Cartesian product of two median graphs is also median [18].

Given an integer $k \geq 1$, we denote by Q_k the hypercube of dimension k. Inductively, graph Q_1 is the single-edge graph, and Q_k is the Cartesian product of Q_{k-1} and Q_1 for $k \geq 2$. In other words, Q_k is obtained by taking graph Q_{k-1} with a copy of it, and connecting each vertex with its own copy. For example, Q_2 is a square and Q_3 is the well-known 8-vertex cube of dimension 3.

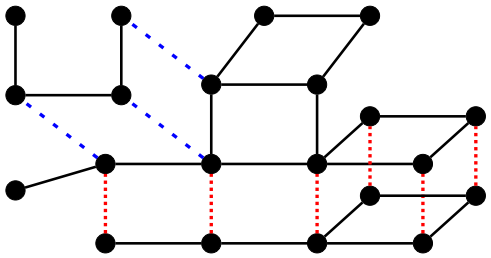

Figure 1: Example of median graph G with $d = 3$. Two Θ-classes are highlighted.

Definition 5 (Dimension d)**.** *The dimension $d = dim(G)$ of a median graph G is the dimension of the largest hypercube contained in G as an induced subgraph.*

In other words, $d = \dim(G)$ means that G admits Q_d as an induced subgraph, but not Q_{d+1}. Median graphs with $d = 1$ are exactly the trees. Median graphs with $d \leq 2$ are called *cube-free* median graphs. Figure 1 provides us with an example of median graph with dimension $d = 3$. In general, since Q_d contains 2^d vertices:

$$d = \dim(G) \leq \log_2 n.$$

As stated previously, in general graphs, all gated subgraphs are convex. In median graphs, these two characterizations are equivalent.

Lemma 3 ([4, 14])**.** *A subgraph of a median graph is gated if and only if it is convex.*

In fact, fibers are also gated.

Lemma 4 ([19, 23])**.** *For any gated set H of a median graph G and any vertex $x \in H$, the set $F_H[x]$ is gated.*

We recall the notion of Θ-class [25] and its implications on median graphs. We say that the edges uv and xy are in relation Θ_0 if there is a square $uvyx$, where uv and xy are opposite. Notation Θ refers to the reflexive and transitive closure of relation Θ_0. The classes of the equivalence relation Θ are denoted by $E_1, \ldots, E_q$ and the set of Θ-classes is $\mathcal{E}(G)$.

Definition 6 (Θ-classes)**.** *Two edges uv and $u'v'$ belong to the same Θ-class if there is a sequence $uv, u_1v_1, \ldots, u_rv_r, u'v'$ such that u_iv_i and $u_{i+1}v_{i+1}$ are opposite edges of a square.*

The size q of $\mathcal{E}(G)$ can fluctuate: hypercube Q_k admits $q = k$ Θ-classes but 2^k vertices, while a tree has $q = n - 1$ Θ-classes (as many as edges). In any case, the Θ-classes can be assigned to edges of a median graph in linear time.

Lemma 5 (Θ-classes in linear time [14])**.** *There exists an algorithm which computes the Θ-classes $E_1, \ldots, E_q$ of a median graph in linear time $O(m) = O(n \log n)$.*

In median graphs, each class E_i, $1 \leq i \leq q$, is a perfect matching cutset and its two sides H'_i and H''_i admit convex characterizations.

Lemma 6 (Halfspaces of E_i [14, 28, 35])**.** *Let G be any median graph and $E_i \in \mathcal{E}(G)$. For any $1 \leq i \leq q$, the graph G deprived of edges of E_i,* i.e. *$G \backslash E_i = (V, E \backslash E_i)$, has two connected components with respective vertex sets H'_i and H''_i, called* halfspaces*. Edges of E_i form a matching. Halfspaces satisfy the following properties:*

- *Both H_i' and H_i'' are convex/gated.*
- *If uv is an edge of E_i with $u \in H_i'$ and $v \in H_i''$, then:*
  $H_i' = \{x \in V : d(x,u) < d(x,v)\}$
  $H_i'' = \{x \in V : d(x,v) < d(x,u)\}$.

We say a *minority halfspace* is a halfspace, say H_i' w.l.o.g, such that $|H_i'| < |H_i''|$. The opposite halfspace H_i'' is thus naturally called a *majority halfspace*. Moreover, we say that H_i' and H_i'' are *egalitarian halfspaces* if $|H_i'| = |H_i''|$. In summary, a pair (H_i', H_i'') is made up of either a minority-majority configuration or two egalitarian halfspaces.

A very nice and powerful characterization of median graphs is the construction by convex expansions proposed by Mulder [34, 35]. Indeed, given a graph G with two convex subgraphs G_1 and G_2 covering G, the *convex expansion* G' of G is the graph obtained by adding both an isomorphic copy of $G_1 \cap G_2$ and a matching connecting the corresponding vertices between $G_1 \cap G_2$ and its copy. While $G_1 \cap G_2$ stays attached to G_1, its copy replaces it in G_2, and the matching connects both versions of $G_1 \cap G_2$. The apparition of this matching in the new graph G' consists in the creation of a new Θ-class. In summary, any median graph can be obtained by successive convex expansions starting from the single-vertex graph [34].

We denote by $\partial H_i'$ the subset of H_i' containing the vertices which are adjacent to a vertex in H_i'': $\partial H_i' = N(H_i'')$. Put differently, the set $\partial H_i'$ is made up of vertices of H_i' which are endpoints of edges in E_i. Symmetrically, set $\partial H_i''$ contains the vertices of H_i'' which are adjacent to H_i'. We say these sets are the *boundaries* of halfspaces H_i' and H_i'' respectively. A halfspace satisfying $H_i' = \partial H_i'$ is called a *peripheral halfspace* and its associated Θ-class is also called a *peripheral Θ-class*. As observed in [14] any median graph necessarily admits at least one peripheral Θ-class.

Lemma 7 (Boundaries [14, 28, 35])**.** *Let G be a median graph and $E_i \in \mathcal{E}(G)$. Both $\partial H_i'$ and $\partial H_i''$ are convex/gated. Also, the edges of E_i define an isomorphism between $\partial H_i'$ and $\partial H_i''$.*

As a consequence, suppose uv and $u'v'$ belong to E_i: if uu' is an edge and belongs to class E_j, then vv' is an edge too and it belongs to E_j. We pursue with another property related to the fact that median graphs G are bipartite. The *v_0-orientation* of the edges of G according to some vertex $v_0 \in V(G)$ is such that, for any edge uv, the orientation is $\overrightarrow{uv}$ if $d(v_0,u) < d(v_0,v)$. Indeed, we cannot have $d(v_0,u) = d(v_0,v)$ as G is bipartite.

Lemma 8 (Orientation [14])**.** *All edges of a median graph G can be oriented according to any vertex $v_0 \in V(G)$.*

Considering such orientation fixed, we might refer to the vertex v_0 as the *canonical base-point*. Given two vertices $u, v \in V$, the set which contains the Θ-classes separating u from v is called the *signature* $\sigma_{u,v}$. For example, if $u \in H_i'$ and $v \in H_i''$, then $E_i \in \sigma_{u,v}$.

Definition 7 (Signature $\sigma_{u,v}$ [15])**.** *We say that the* signature *of the pair of vertices u, v, denoted by $\sigma_{u,v}$, is the set of classes E_i such that u and v are separated in $G \backslash E_i$. In other words, u and v are in different halfspaces of E_i.*

As stated in [16], the signature of two vertices provides us with the composition, in terms of Θ-classes, of any shortest (u,v)-path.

Lemma 9 ([16]). *For any shortest (u, v)-path P, the edges in P belong to classes in $\sigma_{u,v}$ and, for any $E_i \in \sigma_{u,v}$, there is exactly one edge of E_i in path P. Conversely, a path containing at most one edge of each Θ-class is a shortest path between its departure and its arrival.*

This lemma is a consequence of the convexity of halfspaces. Indeed, a shortest path that would pass through two edges of some Θ-class E_i would escape temporarily from an halfspace. This is not possible since any halfspace is convex (Lemma 6).

2.3 Orthogonal Θ-classes, POFs and ladder sets

We now present other notions on median graphs related to the *orthogonality* of Θ-classes [30].

Definition 8 (Orthogonal Θ-classes [30]). *We say that classes E_i and E_j are* orthogonal *(denoted by $E_i \perp E_j$) if there is a square $uvyx$ in G, where $uv, xy \in E_i$ and $ux, vy \in E_j$.*

We focus on the set of Θ-classes which are pairwise orthogonal.

Definition 9 (Pairwise Orthogonal Family (POF) [15]). *We say that a set of classes $X \subseteq \mathcal{E}(G)$ is a* POF *if for any pair $E_j, E_h \in X$, we have $E_j \perp E_h$.*

The empty set is considered as a POF, such as the singletons of elements of $\mathcal{E}(G)$. The notion of POF has natural connections with induced hypercubes in median graphs. As an extreme case, the whole set $\mathcal{E}(G)$ is a POF if and only if graph G is a hypercube of dimension $\log_2 n$ [30, 32]. Conversely, if G does not admit any POF of size at least 2, then it is a tree, as it means that there is no square in G. The following lemma states an important observation linking POFs and hypercubes.

Lemma 10 (POFs and hypercubes [16]). *Let X be a POF, $v \in V(G)$, and assume that for each $E_i \in X$, there is an edge of E_i adjacent to v. There exists a hypercube Q containing vertex v and all edges of X adjacent to v. The Θ-classes of the edges of Q are the Θ-classes of X.*

Furthermore, a natural bijection between the vertices of a median graph and its POFs was highlighted in the literature [5, 11]. As a consequence, the number of POFs is equal to $n = |V(G)|$.

Lemma 11 (POF/vertex bijection [5, 11]). *Let G be a median graph and $v_0 \in V(G)$ an arbitrary basepoint. We consider the v_0-orientation of G. Given a vertex $v \in V(G)$, let $N^-(v)$ be the set of edges going into v and $\mathcal{E}^-(v) \subseteq \mathcal{E}(G)$ the Θ-classes of the edges in $N^-(v)$.*

- *For any vertex $v \in V(G)$, $\mathcal{E}^-(v)$ is a POF. Moreover, both v and the edges of $N^-(v)$ belong to an induced hypercube whose edges are in the Θ-classes of $\mathcal{E}^-(v)$.*
- *For any POF X, there is an unique vertex v_X such that $\mathcal{E}^-(v_X) = X$. Vertex v_X is the closest-to-v_0 vertex v such that $X \subseteq \mathcal{E}^-(v)$. As v_X and $N^-(v_X)$ belong to a common induced hypercube, any POF X verifies $|X| \leq d$.*

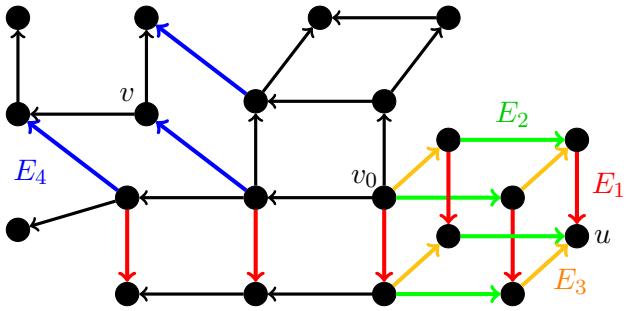

Figure 2: The v_0-orientation of some median graph G and some of its Θ-classes. For example, $\mathcal{E}^-(u) = \{E_1, E_2, E_3\}$.

An example is given in Figure 2 with the same median graph than in Figure 1. The v_0-orientation of this graph is represented, with a sample of four Θ-classes. With the notation used in the previous lemma, we have: $\mathcal{E}^-(v_0) = \emptyset$, $\mathcal{E}^-(v) = \{E_4\}$, and $\mathcal{E}^-(u) = \{E_1, E_2, E_3\}$. One can check that each vertex admits its own incoming set of Θ-classes. Due to this POF/vertex bijection, POFs of a median graph can be enumerated in linear time [11, 30]. Furthermore, as with the v_0-orientation, at most d arcs enter in each vertex ($\mathcal{E}^-(u)$ is a POF), median graphs are relatively sparse: $m \leq dn \leq n \log_2 n$.

We focus on another notion strongly related to POFs, defined in [16], called *ladder set*.

Definition 10 (Ladder set [16])**.** *Given two vertices $u \neq v$ of a median graph G, the ladder set $L_{u,v}$ is the set of Θ-classes which are both adjacent to u and belong to $\sigma_{u,v}$:*

$$L_{u,v} = \left\{ E_i \in \sigma_{u,v} : u \in \partial H_i' \cup \partial H_i'' \right\}.$$

For example, in Figure 2, we have $L_{v,u} = L_{v,v_0} = \{E_4\}$ and $L_{u,v} = L_{v_0,u} = \{E_1, E_2, E_3\}$. In fact, the Θ-classes of a given ladder set are pairwise orthogonal.

Lemma 12 ([16])**.** *Any ladder set $L_{u,v}$ is a POF.*

Less formally, the ladder set $L_{u,v}$ provides us with the Θ-classes of the induced hypercube containing u covered by the set of all shortest (u, v)-paths.

Given a basepoint v_0, all ladder sets $L_{v_0,v}$, with $v \in V(G) \setminus \{v_0\}$ can be enumerated in quasilinear time thanks to a BFS starting at v_0. Such an algorithm is evoked in [15] but is not clearly stated, hence we do so.

Lemma 13 ([15])**.** *Given a basepoint v_0 of some median graph G, the list of all ladder sets $L_{v_0,v}$ with $v \neq v_0$ can be enumerated in quasilinear time $O(n \log^2 n)$.*

Proof. Initialize the queue with the starting vertex v_0. Label all vertices with a set $\texttt{lad}(x) = \emptyset$. Once a vertex v is at the head of the queue, all its not yet discovered neighbors w of v are inserted into the queue, and we fix $\texttt{lad}(w) = \texttt{lad}(v) \cup E_i$ if $vw \in E_i$ and v_0 is adjacent to E_i, otherwise $\texttt{lad}(w) = \texttt{lad}(v)$. Labels $\texttt{lad}$ exactly compute ladder sets from Definition 10: $\texttt{lad}(v)$ contains the Θ-classes adjacent to v_0 which separates v from v_0. As ladder sets are POFs which contain the identity of at most $\log_2 n$ Θ-classes, the size needed to store each label is at most $(\log_2 n)^2$. BFS execution together with the writing of labels gives a total running time $O(n \log^2 n)$ since $m = O(n \log n)$. □

2.4 Median set and majority rule

It was recently proposed in [14] a linear-time algorithm which computes the median set of median graphs. We recall here some key observations of this article but also previous works that will be useful for us. We begin with the definition of a median set.

Definition 11 (Median vertex and set)**.** *Given a graph G, a* median vertex *of G is a vertex u which minimizes $\Gamma(u) = \sum_{v \in V(G)} d(u,v)$. The median set Med($G$) is the set containing all median vertices.*

In median graphs, the median set admits an interesting characterization related to the Θ-classes. Indeed, any vertex which belongs to at least one minority halfspace is not median.

Lemma 14 (Majority rule [3])**.** *Med(G) is the intersection of all majority halfspaces. It coincides with the interval of a diametral pair of its vertices.*

The majority rule is a key tool for the algorithm presented in [14]. The first part consists in computing the cardinality of each halfspace of the input median graph G. This is based on a *peripheral peeling* that we describe briefly. Consider G where all vertex weights are fixed to 1. Pick up some peripheral Θ-class E_i : first retrieve the cardinality of its peripheral halfspace (say $H'_i = \partial H'_i$) by summing up all weights of H'_i, second transfer the weights of vertices in H'_i to their E_i-neighbors. Remove H'_i from the current graph and recurse the process on another peripheral Θ-class.

Lemma 15 (Halfspaces sizes [14])**.** *Given a median graph G, there is a combinatorial algorithm computing in linear time $O(m) = O(n \log n)$ all triplets $(E_i, |H'_i|, |H''_i|)$ where $E_i \in \mathcal{E}(G)$.*

Once the cardinality of each halfspace is known, the majority rule allows them to retrieve the median set Med(G). The second part consists simply in orienting edges uv (say $uv \in E_j$ and $u \in H'_j$ w.l.o.g) such that v is the head of the arc iff $|H''_j| > |H'_j|$. From Lemma 14, the median set coincides with the sinks of this partially directed graph.

Corollary 1 ([14])**.** *Med(G) can be computed in linear time $O(m) = O(n \log n)$.*

3 Exploiting halfspaces for the computation of eccentricities

Our objective is to determine the weighted eccentricities of a weighted median graph (G, ω). Please note that notation G naturally refers to the same graph without any weight consideration. To achieve our goal, we introduce the notion of balanced Θ-classes. Our algorithm will distinguish two cases: either G contains a balanced Θ-class or not. If it is the case, then we will pick up such a Θ-class and retrieve the eccentricities of (G, ω) by computing recursively the eccentricities of the two halfspaces $(G[H'_i], \omega)$ and $(G[H''_i], \omega)$.

3.1 Balanced Θ-classes

As stated in Section 2, the Θ-classes are natural separators for a median graph. Consequently, the ratio between the size of the two halfspaces is a potential tool to design divide-and-conquer procedures on this family of graphs. We begin with the definition of f-balanced Θ-classes.

Definition 12 (Balanced and unbalanced Θ-classes)**.** *Let $\mathbb{N}^* = \mathbb{N} \setminus \{0\}$ be the set of positive integers. Given some median graph G and a function $f : \mathbb{N}^* \to \mathbb{N}^*$, a Θ-class E_i of G is f-balanced if:*

$$\min\{|H_i'|, |H_i''|\} \geq \frac{|V(G)|}{f(|V(G)|)}. \tag{1}$$

Conversely, a Θ-class is f-unbalanced if it is not f-balanced, so formally either $|H_i'| < \frac{|V(G)|}{f(|V(G)|)}$ or $|H_i''| < \frac{|V(G)|}{f(|V(G)|)}$.

The halfspaces of a Θ-class E_i form a bipartition of $V(G)$. Hence, being f-unbalanced means that one halfspace has size less than $\frac{n}{f(n)}$, with $n = |V(G)|$, while the other halfspace is large with cardinality at least $n(1 - \frac{1}{f(n)})$. Star graphs are the most natural examples of median graphs without balanced Θ-classes: for each Θ-class, its minority halfspace has size 1. Observe that the case f being a constant function (for example, $f(n) = 3$ for all integers n) is the classical sense given to a *balanced separator* in graphs, as defined originally by [31]. In the remainder of this article, we will focus more particularly on f-balanceness, with f being a logarithmic function.

It should be noticed that certain median graphs do not admit any f-balanced Θ-classes, for some given function f.

Definition 13 (Unbalanced median graphs)**.** *Let $\mathcal{U}_f$ be the family of all median graphs G satisfying the following property: all Θ-classes of G are f-unbalanced.*

For Sections 3 and 4, we fix a specific function f and the notion of *balanced/unbalanced* Θ-class will be naturally associated with this function : let $f = 2\log$, *i.e.* the function $f : n \to 2\log(n)$. We denote by $\mathcal{U}_{2\log}$ the family of median graphs which do not admit any $(2\log)$-balanced Θ-classes. Fixing $f = 2\log$, which is less restrictive than a constant function, is crucial to make our algorithm work: this choice will be justified in the proof of Theorem 1 (as presented in section 1).

Checking whether a median graph admits (or not) a balanced Θ-class can be achieved in linear time $O(n \log n)$, according to Lemma 15. As a consequence, we are able in linear time either to pick up a balanced Θ-class (for any balance criterion f) or answer that the input median graph contains only unbalanced ones. It consists simply in applying the algorithm of Lemma 15 and returning a balanced Θ-class when Equation (1) is satisfied.

The idea of our future recursive scheme is to identify whether a balanced Θ-class of the input median graph G exists. If it is the case, we compute recursively the weighted eccentricities of its halfspaces in order to retrieve the weighted eccentricities of the whole graph. Otherwise, we fall into the case $G \in \mathcal{U}_{2\log}$ which will be treated in Section 4.

We state an analytical result on some integer sequences which will be a key win-win argument for this global recursive process. The consequence of the following lemma is that the depth of the recursive tree, built upon balanced Θ-classes separation, is at most poly-logarithmic.

Lemma 16. *Let (s_n) be a sequence of integers such that s_0 is a positive integer, and let $\lambda \geq 2$ be a positive real number such that:*

$$s_{n+1} = \begin{cases} \lfloor s_n \left(1 - \frac{1}{\lambda \log(s_n)}\right) \rfloor & \text{if } s_n > 2 \\ s_n & \text{if } s_n \leq 2 \end{cases}$$

The total stopping time τ of sequence s_n, i.e. *the minimum positive integer τ such that $s_\tau = s_{\tau+1}$, verifies $\tau \leq \lambda(\log(s_0))^2$. Moreover, $s_\tau \in \{1, 2\}$.*

Proof. Consider a finite subsequence $s_0, s_1, \ldots, s_n$ and assume that $s_n > 2$. By definition, it is monotonically decreasing since $\lambda \log(s_i) > 1$ for any $0 \leq i \leq n$.

Then, $s_n \leq s_0 \prod_{i=0}^{n-1} \left(1 - \frac{1}{\lambda \log(s_i)}\right) \leq s_0 \left(1 - \frac{1}{\lambda \log(s_n)}\right)^n$. Assume by contradiction that $n \geq \lambda(\log(s_0))^2$, then by exploiting the fact that $(1 - \frac{1}{x})^x \leq \frac{1}{e}$ for $x \geq 2$,

$$s_n \leq s_0 \left(1 - \frac{1}{\lambda \log(s_n)}\right)^{\lambda(\log(s_0))^2} \leq s_0 \left(1 - \frac{1}{\lambda \log(s_n)}\right)^{\lambda \log(s_n) \log(s_0)} \leq s_0 e^{-\log(s_0)} = 1.$$

This yields a contradiction since we assumed $s_n > 2$. Hence, for $n \geq \lambda(\log(s_0))^2$, $s_n \leq 2$, so the total stopping time verifies $\tau \leq \lambda(\log(s_0))^2$. Now, we verify that $s_\tau > 0$. As $s_{\tau-1} \geq 3$, we have $s_\tau = \lfloor s_{\tau-1} \left(1 - \frac{1}{\lambda \log(s_{\tau-1})}\right) \rfloor \geq \lfloor 3(1 - \frac{1}{2\log(3)}) \rfloor \geq 1$. Therefore, $s_\tau \in \{1, 2\}$. □

3.2 Retrieving all eccentricities thanks to Θ-classes

The following theorem is crucial for our algorithm: it states that, given a Θ-class and the weighted eccentricity of each vertex inside its induced halfspace, we can re-assemble all weighted eccentricities of (G, ω) in linear time.

Theorem 3. *Let (G, ω) be a weighted median graph and E_i one of its Θ-classes. Assume that:*

- *all weighted eccentricities of $(G[H_i'], \omega)$ are known,*
- *all weighted eccentricities of $(G[H_i''], \omega)$ are known.*

Then, one can compute all weighted eccentricities of (G, ω) in linear time $O(|E(G)|) = O(n \log n)$.

Proof. At the beginning of our computation, every vertex of $V(G)$ is labeled with a weighted distance: the vertices $u' \in H_i'$ are labeled with their weighted eccentricity in $G[H_i']$, formally $\mathsf{ecc}\,(u' \mid (H_i', \omega))$. Similarly, each vertex $u'' \in H_i''$ is labeled with $\mathsf{ecc}\,(u'' \mid (H_i'', \omega))$.

For any vertex $u' \in H_i'$, we determine its H_i''-gate $g(u') \in H_i''$ and, conversely, for any vertex $u'' \in H_i''$, we determine its H_i'-gate $g(u'')$. This operation can be achieved in total $O(n \log n)$ time, as recalled in Lemma 2, by launching two BFSs: one with a starting queue made up of H_i', and one with a starting queue H_i''. We compute, for any $v' \in \partial H_i'$ (resp. $v'' \in \partial H_i''$) its open fiber $F_{H_i'}(v')$ in H_i'' (resp. $F_{H_i''}(v'')$ in H_i'). Through this BFS, we store the unweighted distance from any vertex to its gate: we obtain all values $d(u', g(u'))$ (resp. $d(u'', g(u''))$ also in linear time.

One can retrieve at this moment the weighted eccentricity of $u' \in H_i'$ (resp. $u'' \in H_i''$ with the same arguments). Indeed, the farthest vertex from u' (in the weighted sense) is either in H_i' or in H_i''. If it belongs to H_i', as this halfspace is convex, the weighted eccentricity of u' is its label $\mathsf{ecc}\,(u' \mid (H_i', \omega))$. Else, as H_i'' is gated, for any vertex $v'' \in H_i''$, there is a shortest path from u' to v'' passing through the H_i''-gate $g(u')$. Conversely, any shortest path induced in H_i'' from $g(u')$ to some v'', concatenated with a shortest $(u', g(u'))$-path, produces a shortest (u', v'')-path, since $d(u', g(u')) + d(g(u'), v'') = d(u', v'')$. Hence, the weighted eccentricity of

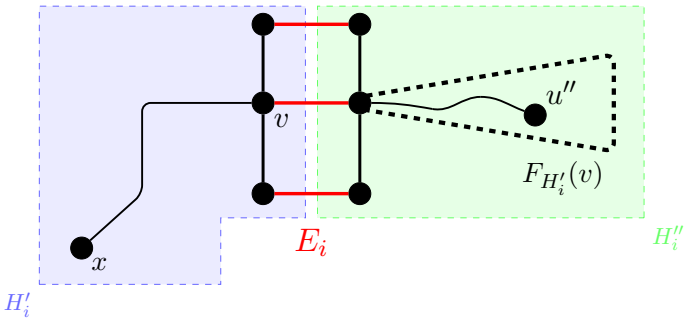

Figure 3: The largest weighted distance from $u'' \in H''_i$ to some vertex $x \in H'_i$ is given by the (unweighted) distance from u'' to its gate $v = g_{H'_i}(u'')$ in addition with the weighted eccentricity of v: $\mathsf{ecc}\,(v \mid (H'_i, \omega)) = d(v, x)$.

u' can be decomposed as the sum of $d(u', g(u'))$ (distance to the gate) with the weighted eccentricity of $g(u')$ in H''_i. Formally,

$$\mathsf{ecc}\left(u' \mid (G, \omega)\right) = \max\left\{\mathsf{ecc}\left(u' \mid (H'_i, \omega)\right),\ d(u', g(u')) + \mathsf{ecc}\left(g(u') \mid (H''_i, \omega)\right)\right\}.$$

Figure 3 illustrates this formula with a vertex $u'' \in H''_i$ instead of $u' \in H'_i$.

The whole procedure, consisting first in computing the gates $g(u)$ and second in applying the latter formula, takes time $O(n \log n)$. □

Theorem 3 immediately yields a recursive algorithm scheme based on Θ-classes. But its complexity is not necessarily subquadratic.

When our input graph G has a balanced Θ-class E_i, one can recursively compute the weighted eccentricities of the two induced halfspaces $G[H'_i]$ and $G[H''_i]$ and then retrieve the weighted eccentricities of G with an extra linear time. With some "ideal" instance, we could always find such a balanced Θ-class and use this recursive scheme until falling into the trivial case of a single weighted vertex. Observe that, with such a utopian situation, the total running time would be quasilinear: the depth of the recursive tree is poly-logarithmic thanks to Lemma 16. Indeed, fixing $s_0 = n = |V(G)|$ and $\lambda = 2$, value s_n gives an upper bound of the number of vertices remaining in each branch of the recursive tree after n calls.

Nevertheless, we might find at some moment an induced subgraph of G which has no balanced Θ-class, in other words which belongs to $\mathcal{U}_{2\log}$. Obviously, even the input G could belong to this family. Then the time complexity of the recursive scheme could be quadratic.

As a conclusion of this section, we observe that, to pursue the description of our recursive algorithm, we need to focus on the case of weighted median graphs which admit only unbalanced Θ-classes.

4 Computation of eccentricities on median graphs without balanced Θ-classes

The study of median graphs with only unbalanced Θ-classes is of interest since, as we prove it now, the existence of a quasilinear-time algorithm finding all weighted eccentricities for the subfamily $\mathcal{U}_{2\log}$ of median graphs will imply the same result for the whole family of median graphs. In Sections 4.1 and 4.2, we fix $(G, \omega) \in \mathcal{U}_{2\log}$.

4.1 Slice decomposition

We now pursue with new concepts about Θ-classes. As all Θ-classes E_i of $G \in \mathcal{U}_{2\log}$ are unbalanced, all of them admit both a minority halfspace denoted from now on by H_i' (the halfspace with the smaller size), and a majority halfspace denoted by H_i''. From Definition 12, we have $|H_i'| \leq \frac{n}{2\log(n)}$ with $n = |V(G)|$. Also, we assume that graph $G \in \mathcal{U}_{2\log}$ has at least three vertices: $n \geq 3$ and $\log n \geq 1$ (case $n \leq 2$ will be treated trivially).

No graph of $\mathcal{U}_{2\log}$ contains an egalitarian halfspace since all its Θ-classes are unbalanced. Consequently, these graphs admit a unique median vertex.

Lemma 17 (Unique median vertex in $G \in \mathcal{U}_f$)**.** *For any n-vertex median graph $G \in \mathcal{U}_f$ where $f(n) \geq 2$, there exists a unique vertex v_0 such that, for any Θ-class E_i, this vertex v_0 belongs to the majority halfspace of E_i.*

Proof. From Lemma 14, the median set of a graph $G \in \mathcal{U}_f$ is the intersection of all majority halfspaces and there exists $u, v \in V(G)$ such that $\mathrm{Med}(G) = I(u, v)$. Assume that $u \neq v$. As $\mathrm{Med}(G)$ is connected, consider two adjacent vertices $x, y \in \mathrm{Med}(G)$ and let $xy \in E_i$. The Θ-class E_i admits two egalitarian halfspaces since, otherwise, either x or y would not be a median vertex. We have a contradiction: E_i is clearly balanced, but it should admit a halfspace of size strictly smaller than $\frac{n}{f(n)} \leq \frac{n}{2}$. □

From now on, this unique median vertex v_0 will be taken as the basis of the orientation of G. We consider a second vertex $u_{\max}$ (potentially equal to v_0) which is a farthest vertex from v_0.

Definition 14. *Let $u_{\max}$ be a vertex such that $d_\omega(v_0, u_{\max}) = \mathsf{ecc}(v_0 \mid (G, \omega))$. If several candidates for $u_{\max}$ exist, then select one arbitrarily, except in one case: when v_0 itself is a candidate for $u_{\max}$, then fix $u_{\max} = v_0$.*

We begin with a straightforward observation: if $u_{\max} = v_0$, then v_0 is the farthest vertex from any $u \neq v_0$.

Lemma 18. *If $v_0 = u_{\max}$, then for any vertex $u \in V(G) \setminus \{v_0\}$, $\mathsf{ecc}(u \mid (G, \omega)) = d_\omega(u, v_0)$.*

Proof. By definition of $u_{\max}$, we have $\omega(v_0) \geq d_\omega(v_0, u)$. Consider any vertex $v \neq v_0$. By triangular inequality, $d(u, v) \leq d(u, v_0) + d(v_0, v)$, so $d_\omega(u, v) \leq d(u, v_0) + d(v_0, v) + \omega(v) = d(u, v_0) + d_\omega(v_0, v)$. But $d_\omega(v_0, v) \leq \omega(v_0)$ which implies that $d_\omega(u, v) \leq d_\omega(u, v_0)$. As a consequence, v_0 is the farthest vertex from any $u \neq v_0$. □

A natural consequence of the previous lemma is that, if $u_{\max} = v_0$, then the eccentricities of $(G, \omega) \in \mathcal{U}_{2\log}$ can be computed in linear time in a very simple way.

Corollary 2. *Let $(G, \omega) \in \mathcal{U}_{2\log}$ with $u_{\max} = v_0$. All eccentricities of (G, ω) can be computed in linear time $O(m) = O(n \log n)$ thanks to a BFS starting at v_0.*

Proof. Executing a BFS with departure vertex v_0 allows us to obtain all unweighted distances between v_0 and all other vertices. We know from Lemma 18 that, in our case, for any $u \neq v_0$, $\mathsf{ecc}(u \mid (G, \omega)) = d_\omega(u, v_0) = d(u, v_0) + \omega(v_0)$. Moreover, $\mathsf{ecc}(v_0 \mid (G, \omega)) = \omega(v_0)$ since $u_{\max} = v_0$. In summary, after computing all unweighted distances from v_0 in linear time, one can retrieve all eccentricities of (G, ω) also in linear time as a second step. □

From now on, we can assume that $u_{\max} \neq v_0$ as otherwise the weighted eccentricities can be computed trivially. We define $L(G)$ as a ladder set $L_{v_0,u_{\max}}$ between v_0 and one farthest-to-v_0 vertex $u_{\max}$.

Definition 15 (Wide ladder)**.** *We denote by $L(G)$ the* wide ladder *of G, which is the POF containing the Θ-classes E_i adjacent to v_0 such that $u_{\max}$ belongs to their minority halfspace,* i.e. *$u_{\max} \in H'_i$. Formally, $L(G) = L_{v_0,u_{\max}}$ and $\ell = |L(G)|$ is its size.*

For sake of simplicity, we modify the indices of the Θ-classes such that:

$$L(G) = \{E_1, E_2, \ldots, E_\ell\}.$$

Observe that this change has no impact on the previous results since the indices of Θ-classes played no role yet.

Our technique to handle median graphs without balanced Θ-classes relies on the following crucial observation. All vertices v of G such that $L_{v_0,v} \cap L(G) = \emptyset$ form, due to the unbalancedness of the Θ-classes, a large set of vertices. Moreover, we can directly deduce their weighted eccentricity which is $d_\omega(v, u_{\max})$. As a consequence, the eccentricity of at least half of the vertices of the graph are already known. We will explain how to handle the remaining vertices afterwards, in Section 4.2.

Lemma 19. *Let $G \in \mathcal{U}_{2\log}$ and v_0 its median vertex. Any vertex v such that $L_{v_0,v} \cap L(G) = \emptyset$ has a weighted eccentricity $\mathsf{ecc}(v \mid (G,\omega)) = d(v, v_0) + d_\omega(v_0, u_{\max}) = d_\omega(v, u_{\max})$.*

Proof. It suffices to show that $v_0 \in I(v, u_{\max})$. Let P be a $(v, u_{\max})$-path consisting in the concatenation of a shortest (v, v_0)-path (denoted by Q) with a shortest $(v_0, u_{\max})$-path (denoted by R). It suffices to show that P is made up of edges belonging to pairwise different Θ-classes (Lemma 9). Assume by contradiction that there exists a Θ-class $E_j \in \sigma_{v_0,v} \cap \sigma_{v_0,u_{\max}}$ and that there is no other Θ-class $E_k \in \sigma_{v_0,v} \cap \sigma_{v_0,u_{\max}}$ such that:

- the edge of E_k on path Q is closer to v_0 than the edge of E_j on Q,
- the edge of E_k on path R is closer to v_0 than the edge of E_j on R.

In brief, if we assume $\sigma_{v_0,v} \cap \sigma_{v_0,u_{\max}}$ nonempty, we fix E_j as a Θ-class of this set which is minimal by distance towards the median vertex v_0. Such a minimal class necessarily exists.

By definition of E_j, there is no Θ-class that appears twice in the path P_j, defined as the connected sub-path containing v_0 obtained after removing edges of E_j in the path $G[P]$. Hence, P_j is a shortest path. Moreover, its endpoints both belong to the same boundary $\partial H''_j$ of E_j. As $\partial H''_j$ is convex (Lemma 7) and $v_0 \in P_j$, then $v_0 \in \partial H''_j$. This yields a contradiction since E_j is adjacent to v_0 and should be part of $L_{v_0,v} \cap L(G)$. □

Definition 16 (Large sets)**.** *We define recursively a finite sequence $(G_i)_{0 \leq i \leq \ell}$ of induced subgraphs of G called* large sets*. Let $G_0 = G$ and, for any integer $0 \leq i \leq \ell - 1$,*

$$G_{i+1} = G_i \setminus H'_{i+1}.$$

In other words, G_{i+1} is incrementally obtained from G_i by removing all vertices in the minority halfspace of $E_{i+1} \in L(G)$.

A first trivial remark is that the number of vertices of the graphs of the sequence is decreasing: $|V(G_{i+1})| \leq |V(G_i)|$. A second fact is that each G_i is a gated subgraph of G.

Lemma 20. *For each $0 \leq i \leq \ell$, G_i is a gated subgraph of G.*

Proof. We proceed by induction. Obviously, as the base case, $G_0 = G$ is trivially gated.

We assume that G_i is a gated subgraph of G: $V(G_i)$ is gated in G. Moreover, the halfspace H''_{i+1} is also gated in G (Lemma 6). As we know from Definition 16, $V(G_{i+1}) = V(G_i) \backslash H'_{i+1} = V(G_i) \cap H''_{i+1}$. The intersection of two gated sets is gated (Lemma 1). Hence, $V(G_{i+1})$ is gated and our induction step holds. □

In fact, we can rewrite, for any $1 \leq i \leq \ell$, the definition of G_i as the following one:

$$G_i = G[H''_1 \cap H''_2 \cap \ldots \cap H''_i]. \tag{2}$$

As all large sets G_i are convex/gated subgraphs of G, they are median graphs. So, they admit Θ-classes which can in fact be retrieved from the Θ-classes of the original graph G.

Lemma 21 (Θ-classes of isometric subgraphs [15])**.** *Let G be a median graph. If H is an isometric[1] subgraph of G, then the Θ-classes of H are exactly the nonempty subsets among $E_i \cap E(H)$, for $E_i \in \mathcal{E}(G)$.*

Since gated subgraphs are also isometric (the converse is false), then the Θ-classes of G_i are inherited from the Θ-classes of G.

We define *slices* as the subgraphs withdrawn from each transition between G_i and G_{i+1}.

Definition 17 (Slices)**.** *We define a finite sequence $(S_i)_{0 \leq i \leq \ell-1}$ of graphs called* slices*:*

$$S_i = G_i \setminus V(G_{i+1}).$$

The vertex set of each G_i can be thus partitioned in 2 parts $V(S_i)$ and $V(G_{i+1})$.

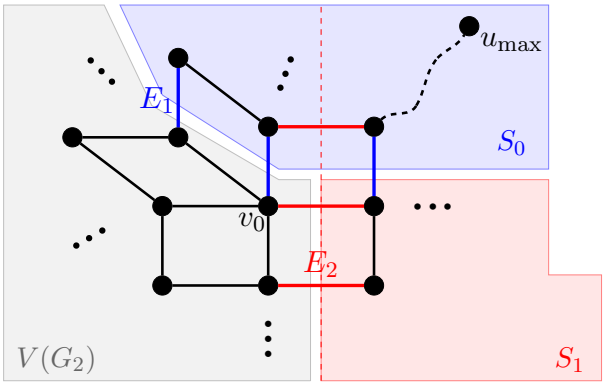

Figure 4: Illustration of the slice decomposition of some median graph G with $L(G) = \{E_1, E_2\}$. For example, $V(G_1) = V(S_1) \cup V(G_2)$.

We refer to the expression *slice decomposition* in order to refer to the whole collection of slices. Slices S_i can be rewritten as the intersection of the minority halfspace H'_{i+1} with

[1] a subgraph which preserves the distances of the original graph

the current vertex set $V(G_i)$: $S_i = G_i[H'_{i+1}]$. Hence, by definition, slices are pairwise vertex-disjoint. In fact,

$$S_i = G\left[H'_{i+1} \setminus \left(H'_1 \cup H'_2 \cup \ldots H'_i\right)\right]. \tag{3}$$

Figure 4 gives an illustration of the slice decomposition on some example. It represents a part of the graph: only $u_{\max}$ together with the vertices belonging to hypercubes which contain the median vertex v_0 are drawn. The wide ladder (ladder set $L_{v_0,u_{\max}}$) contains two Θ-classes E_1 and E_2: $\ell = 2$. In such case, there are two slices: S_0 is exactly the minority halfspace H'_1 and S_1 is H'_2 deprived of the vertices of H'_1: $S_1 = G[H'_2 \setminus H'_1]$. Sets S_0, S_1, and $V(G_2)$ are disjoint and cover the graph.

Lemma 22. *For each $0 \leq i \leq \ell - 1$, S_i is a gated subgraph of G*

Proof. Graph G_i is gated subgraph of G (Lemma 20). Consequently, S_i is gated since it is the intersection between H'_{i+1} and $V(G_i)$ (Lemma 1). □

In terms of cardinality, slices are thus small, $|V(S_i)| \leq |H'_{i+1}| \leq \frac{n}{2\log(n)}$, while even the last large set G_ℓ contains at least half of the vertices.

Lemma 23 (Cardinality of large sets)**.** *For any $0 \leq i \leq \ell$, $|V(G_i)| \geq n(1 - \frac{i}{2\log(n)})$.*

Proof. By induction: $i = 0$ trivially holds. The large set G_{i+1} corresponds to G_i after withdrawing the slice S_i, which contains at most $\frac{n}{2\log n}$ vertices. Therefore by induction,

$$|V(G_{i+1})| \geq |V(G_i)| - \frac{n}{2\log n} = n\left(1 - \frac{i}{2\log(n)}\right) - \frac{n}{2\log n} = n\left(1 - \frac{i+1}{2\log(n)}\right).$$

Which yields the announced inequality for all integers $0 \leq i \leq \ell - 1$. □

Indeed, as $\ell \leq d \leq \log n$, we have $|V(G_\ell)| \geq n(1 - \frac{\ell}{2\log(n)}) \geq \frac{n}{2}$. From Equation (2), set $V(G_\ell)$ contains exactly the vertices which does not belong to any $H'_1, \ldots H'_\ell$, in other words whose ladder set has no intersection with $L(G)$. As stated in Lemma 19, the weighted eccentricities of all vertices v of $V(G_\ell)$ can be directly deduced. On the other hand, any vertex v with a ladder set $L_{v_0,v}$ intersecting $L(G)$ belongs to one of the slices $S_0, \ldots, S_{\ell-1}$ (Equation (3)). The input graph, by definition of slices and large sets, satisfies $V(G) = V(G_\ell) \cup V(S_0) \cup V(S_1) \cup \ldots \cup V(S_{\ell-1})$. Furthermore, the large sets generally satisfy, for any $0 \leq i \leq \ell - 1$:

$$V(G_i) = V(G_\ell) \cup V(S_i) \cup V(S_{i+1}) \cup \ldots \cup V(S_{\ell-1}).$$

4.2 Peeling the slices

We show that, assuming we know all the weighted eccentricities on slices, which are small-sized induced subgraphs of the input graph $(G, \omega) \in \mathcal{U}_{2\log}$, we can deduce all weighted eccentricities of G. We begin with a description of how to modify weights on slices so that our recursive calls will enable us to retrieve all eccentricities of (G, ω).

Definition 18. *Let $(G, \omega) \in \mathcal{U}_{2\log}$ with $u_{\max} \neq v_0$ and an integer $0 \leq i \leq \ell - 1$. The weight function ω_i^* on slice S_i is defined as:*

- *for any* $x \in \partial H'_{i+1} \cap V(S_i)$, $\omega_i^*(x)$ *is equal to the maximum distance* $d_\omega(x,z)$ *where* z *belongs to the fiber* $F_{H'_{i+1}}(x)$ *of* G_i.[2]
- *for any* $y \in V(S_i) \setminus \partial H'_{i+1}$, $\omega_i^*(y) = \omega(y)$.

We state now a theorem which introduces our recursive process to compute the weighted eccentricities of $(G,\omega) \in \mathcal{U}_{2\log}$. As we already observed it, all weighted eccentricities of the large set G_ℓ are already known. Therefore, we focus on computing the eccentricities of the vertices which belong to at least one slice.

Let us now define a labeling function $\mathcal{B}_i$, for any $0 \leq i \leq \ell - 1$ on the vertices of slices $S_i \cup S_{i+1} \cup \ldots \cup S_{\ell-1}$. The label of $\mathcal{B}_i$ for some vertex u is denoted by $\mathcal{B}_i(u)$.

Definition 19 (Labeling $\mathcal{B}_i$)**.** *For any* $0 \leq i \leq \ell - 1$ *and any vertex* $u \in V(S_i) \cup V(S_{i+1}) \cup \ldots \cup V(S_{\ell-1})$*, label* $\mathcal{B}_i(u)$ *is equal to the weighted eccentricity of* u *on graph* (G_i,ω)*:*

$$\mathcal{B}_i(u) = \mathsf{ecc}\left(x \mid (G_i,\omega)\right).$$

Observe that labeling $\mathcal{B}_0$ would allow us to determine all eccentricities of (G,ω): for vertices in the slices, the labels give their eccentricity while for vertices in G_ℓ, we can use Lemma 19.

Our proposition is the following. Assume that we already computed some labeling $\mathcal{B}_{i+1}$ and our intention is to obtain $\mathcal{B}_i$. Our idea to achieve this recursive step consists in launching two recursive calls, in order to compute the weighted eccentricities on graph S_i but with two different weight functions ω and ω_i^*. We prove that given labeling $\mathcal{B}_{i+1}$ and the results of these calls, we can retrieve labeling $\mathcal{B}_i$.

Theorem 4. *Let* $0 \leq i \leq \ell - 1$*. Assume that the following values are known:*

- *all weighted eccentricities of* (S_i,ω)*,*
- *all weighted eccentricities of* (S_i,ω_i^*)*,*
- *all labels of* $\mathcal{B}_{i+1}$*.*

One can compute in linear time $O(|E(G_i)|)$ *all labels of* $\mathcal{B}_i$*.*

Proof. We apply the BFS traversal evoked in Lemma 2 for graph G_i with the gated set S_i. We recall that the vertex set of G_i can be partitioned into $V(G_{i+1})$ and $V(S_i)$. We thus obtain in time $O(|E(G_i)|)$ all the unweighted distances $d(g_{S_i}(v),v)$, for each $v \in V(G_{i+1})$, and as a consequence, also all weighted distances $d_\omega(g_{S_i}(v),v) = d(g_{S_i}(v),v) + \omega(v)$. Our objective is to compute the labels $\mathcal{B}_i(x)$. We distinguish two cases: either x belongs to the current slice S_i, or in a former slice $S_{i+1} \cup \ldots \cup S_{\ell-1}$.

Case 1: $x \in V(S_i)$. We determine the labels $\mathcal{B}_i(x)$ for vertices $x \in S_i$ which are from Definition 19 their eccentricity in (G_i,ω). The weighted distance between some $x \in S_i$ and $v \in V(G_{i+1})$ is:

$$d_\omega(x,v) = d(x,g_{S_i}(v)) + d_\omega(g_{S_i}(v),v) = d(x,g_{S_i}(v)) + d(g_{S_i}(v),v) + \omega(v). \tag{4}$$

[2]Here, we abuse notation: H'_{i+1} refers to the intersection of $V(G_i)$ with H'_{i+1}, which is a halfspace of the Θ-class $E_{i+1} \cap E(G_i)$ (Lemma 21).

Let us recall that, from Definition 18, when x belongs to the boundary of S_i, $\omega_i^*(x)$ is equal to the maximum $d_\omega(x,v)$ such that $v \in F_{S_i}(x)$ - or, said differently, $x = g_{S_i}(v)$. Otherwise, $\omega_i^*(x) = \omega(x)$.

We consider two cases for vertex $x \in V(S_i)$. If $x \in \partial H'_{i+1}$, its weighted eccentricity is achieved:

- either with a vertex z of S_i: $d_\omega(x,z) = \mathsf{ecc}\,(x \mid (S_i,\omega))$, known by the assumption on (S_i,ω),
- or with a vertex z of $F_{S_i}(x)$: $d_\omega(x,z) = \omega_i^*(x)$, computed with the BFS traversal,
- or with a vertex z of $V(G_{i+1})$ not in the fiber $F_{S_i}(x)$: in this case, it will be given by $d_\omega(x,z) = d(x, g_{S_i}(z)) + \omega_i^*(g_{S_i}(z))$.

If the weighted eccentricity of x is attained for some vertex $z \in V(G_{i+1})$ (which corresponds to the two latter bullets), its value is equal to $\mathsf{ecc}\,(x \mid (S_i,\omega_i^*))$. Written briefly,

$$\mathsf{ecc}\,(x \mid (G_i,\omega)) = \max\left\{\mathsf{ecc}\,(x \mid (S_i,\omega))\,,\ \mathsf{ecc}\,(x \mid (S_i,\omega_i^*))\right\}. \tag{5}$$

From the statement assumptions, all the values needed to compute $\mathsf{ecc}\,(x \mid (G_i,\omega))$ in Equation (5) are supposed to be known.

If $x \in V(S_i) \setminus \partial H'_{i+1}$, the reasoning is relatively similar: either the weighted eccentricity of x is achieved either with some $v \in S_i$, or with $v \in V(G_{i+1})$. In the latter case, from Equation (4), the weighted eccentricity of x is $d(x, g_{S_i}(v)) + \omega_i^*(g_{S_i}(v)) = d_{\omega_i^*}(x, g_{S_i}(v))$. As the weight difference between ω_i^* and ω lies only on the boundary, we have: $\mathsf{ecc}\,(x \mid (G_i,\omega)) = \max\left\{\mathsf{ecc}\,(x \mid (S_i,\omega))\,,\ \mathsf{ecc}\,(x \mid (S_i,\omega_i^*))\right\}$. Figure 5 illustrates these two possibilities on the example already introduced in Figure 4: either the maximum weighted distance from x goes towards S_i or towards $V(G_{i+1})$. We completed the first part of the proof: the weighted eccentricities of all $x \in S_i$ in graph (G_i,ω) can be directly deduced from the theorem assumptions: $\mathcal{B}_i(x) = \mathsf{ecc}\,(x \mid (G_i,\omega))$.

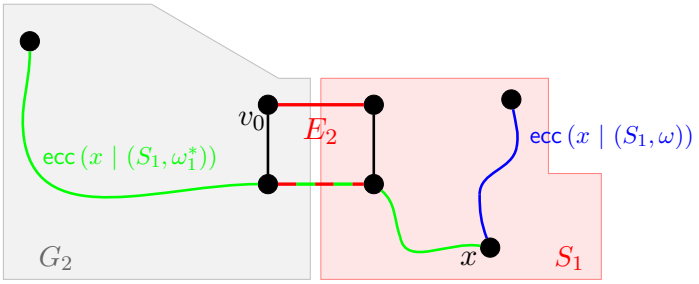

Figure 5: Retrieving the eccentricities of $x \in V(S_i)$ in graph G_i: example with $\ell = 2$ and $i = 1$. Either the largest weighted distance from x is with a vertex of S_i, or with one in the other side $V(G_{i+1})$.

Case 2: $x \in V(S_{i+1}) \cup \ldots \cup V(S_{\ell-1})$. We determine the labels $\mathcal{B}_i(x)$ for vertices $x \in V(S_{i+1}) \cup \ldots \cup V(S_{\ell-1})$. Since we know all labels of $\mathcal{B}_{i+1}$, we have the eccentricity $\mathcal{B}_{i+1}(x) = \mathsf{ecc}\,(x \mid (G_{i+1},\omega))$ of x in graph (G_{i+1},ω). By definition, $\mathcal{B}_i(x) \geq \mathcal{B}_{i+1}(x)$ and the only chance to have $\mathcal{B}_i(x) > \mathcal{B}_{i+1}(x)$ is that the weighted eccentricity of x in G_i is the distance from x to some vertex $v \in V(S_i)$, as $V(G_i) \setminus V(G_{i+1}) = V(S_i)$. So, we compute the maximum distance

$d_\omega(x,v)$ for $v \in V(S_i)$ and verify whether it is larger than $\mathcal{B}_{i+1}(x)$ or not. Thanks to the BFS traversal, we know $g_{S_i}(x)$ and then the maximum distance $D(x) = \max\{d_\omega(x,v) : v \in V(S_i)\}$ is given:

$$D(x) = d(x, g_{S_i}(x)) + \mathsf{ecc}\left(g_{S_i}(x) \mid (S_i, \omega)\right).$$

In summary, all these values can be computed since not only $d(x, g_{S_i}(x))$ but also all eccentricities of (S_i, ω) are known:

$$\mathcal{B}_i(x) = \max\left\{\mathcal{B}_{i+1}(x),\ d(x, g_{S_i}(x)) + \mathsf{ecc}\left(g_{S_i}(x) \mid (S_i, \omega)\right)\right\}$$

. □

The latter theorem stands as the recursive step to prove that the weighted eccentricities of $(G,\omega) \in \mathcal{U}_{2\log}$ can be computed in linear time, assuming that we already have the weighted eccentricities of all slices.

Corollary 3. *Let $(G,\omega) \in \mathcal{U}_{2\log}$ with $u_{\max} \neq v_0$ given with its slice decomposition. Assume that, for each integer $0 \leq i \leq \ell - 1$, all weighted eccentricities of (S_i,ω) but also all weighted eccentricities of (S_i,ω_i^*) are known. Then, one can compute all weighted eccentricities of (G,ω) in time $O(n\log^2 n)$.*

Proof. We proceed by induction in order to prove that labeling $\mathcal{B}_i$ can be determined in time $O((\ell - i)m)$. The base case is the computation of labeling $\mathcal{B}_{\ell-1}$. We focus on graph $G_{\ell-1}$ which is partitioned by Θ-class E_ℓ into two halfspaces: $S_{\ell-1}$ and $V(G_\ell)$. The idea is similar to the ones explained in the proof of Theorem 4. We begin by applying the BFS traversal of Lemma 2 in order to retrieve the $S_{\ell-1}$-gates of any vertex of graph $G_{\ell-1}$. For $x \in S_{\ell-1}$, its weighted eccentricity in $G_{\ell-1}$ is given by:

$$\mathcal{B}_{\ell-1}(x) = \max\{\mathsf{ecc}\left(x \mid (S_{\ell-1},\omega)\right),\ \mathsf{ecc}\left(x \mid (S_{\ell-1},\omega^*_{\ell-1})\right)\}.$$

The knowledge of the eccentricities of $(S_{\ell-1},\omega)$ and $(S_{\ell-1},\omega^*_{\ell-1})$, but also of all distances to $S_{\ell-1}$-gates in $G_{\ell-1}$ thanks to the BFS, allows us to obtain the labeling $\mathcal{B}_{\ell-1}$. The running time is made up of the BFS traversal with a linear number of maximum computations, which is $O(m)$ and independent from ℓ.

The inductive step consists in applying Theorem 4. Indeed, labeling $\mathcal{B}_i$ can be determined from labeling $\mathcal{B}_{i+1}$ in time $O(m)$, given that we have the eccentricities of both (S_i,ω) and (S_i,ω_i^*). Hence, from the induction hypothesis, the running time of the whole process is $O((\ell - i - 1)m + m) = O((\ell - i)m)$.

Consequently, labeling $\mathcal{B}_0$ is obtained in time $O(dm) = O(n\log^2 n)$, as $\ell \leq d \leq \log_2 n$. For any vertex x belonging to a slice, *i.e.* $x \in S_0 \cup \ldots \cup S_{\ell-1}$, value $\mathcal{B}_0(x)$ is its weighted eccentricity on $(G_0,\omega) = (G,\omega)$. Concerning vertices $z \in V(G_\ell)$, as observed earlier, their ladder set $L_{v_0,z}$ has no intersection with $L(G)$, so their weighted eccentricity can be directly computed (Lemma 19). All in all, the weighted eccentricities of (G,ω) are known. □

4.3 Summary and analysis for the eccentricities computation

Given the observations and subroutines presented above, we present now our recursive algorithm which computes all weighted eccentricities of a median graph (G,ω). We call this algorithm MΘRSE (acronym for Median Θ-class Recursive Scheme for Eccentricities) and its

Algorithm 1: Algorithm MΘRSE

1: **Input:** A weighted median graph (G, ω).
2: **Output:** All eccentricities $\mathsf{ecc}(u \mid (G, \omega))$ for each $u \in V(G)$.
3: **if** $|V(G)| \leq 2$ **then**
4: | **return** all eccentricities by enumerating all (at most 2) shortest paths
5: **endif**
6: Compute all Θ-classes $E_i \in \mathcal{E}(G)$ (Lemma 5) and their halfspaces sizes (Lemma 6)
7: **if** *there exists a* Θ*-class* E_i *which is balanced* **then**
8: | LIST1 $\leftarrow$ MΘRSE$(G[H_i'], \omega)$
9: | LIST2 $\leftarrow$ MΘRSE$(G[H_i''], \omega)$
10: | **return** all eccentricities of (G, ω) retrieved from LIST1/LIST2 (Theorem 3)
else
11: | Compute the unique median vertex v_0 as $G \in \mathcal{U}_{2\log}$ (Lemma 17)
12: | Launch a BFS starting from v_0 to determine all distances from v_0, but also $u_{\max}$ and $L(G)$
13: | **if** $u_{\max} = v_0$ **then return** all eccentricities retrieved with Lemma 18
14: | Give a unique (arbitrary) identifier for each class of $L(G)$, from 1 to $\ell = |L(G)|$
15: | Determine all large sets G_j and slices S_i for each $0 \leq j \leq \ell$ and $0 \leq i \leq \ell - 1$
16: | COLLEC $\leftarrow$ list of ℓ empty lists; WCOLLEC $\leftarrow$ list of ℓ empty lists;
17: | **for** i *from 0 to* $\ell - 1$ **do**
18: | | COLLEC$[i]$ $\leftarrow$ MΘRSE(S_i, ω)
19: | | WCOLLEC$[i]$ $\leftarrow$ MΘRSE(S_i, ω_i^*)
20: | **endfor**
21: | **return** all eccentricities of (G, ω) retrieved from COLLEC and WCOLLEC (Theorem 4)
22: **endif**

pseudocode is given in Algorithm 1. The base case of MΘRSE is when the number of vertices of the graph is at most 2. Trivially, one can compute the weighted eccentricites of such graph in constant time $O(1)$ (line 4 of Algorithm 1). We focus now on the recursive calls launched by MΘRSE.

It starts with the computation of Θ-classes and their halfspaces sizes (line 6) which can be done in time $O(n \log n)$, according to Lemmas 5 and 6. Then, it distinguishes two cases: either the input graph (G, ω) contains at least one f-balanced Θ-class (with $f = 2\log$) or not.

If (G, ω) admits a balanced Θ-class E_i, then the procedure is relatively simple. We call recursively MΘRSE in order to obtain the weighted eccentricities of both halfspaces, *i.e.* of $G[H_i']$ and $G[H_i'']$ (lines 8-9). Thanks to Theorem 3, we know that we can retrieve all eccentricities of (G, ω) in linear time (line 10).

Otherwise, $(G, \omega) \in \mathcal{U}_{2\log}$, and there is a unique median vertex v_0 which belong to all majoritarian halfspaces of G (Lemma 17). It is computed in linear time (Corollary 1). Then, we execute a BFS from v_0 (line 12) which allows us to determine $u_{\max}$ and $L(G)$, according to Lemma 13. The slice decomposition of G can be directly deduced from this BFS since the vertices of S_i are exactly the vertices v such that $E_{i+1} \in L_{v_0,v}$ but $E_1, \ldots, E_i \notin L_{v_0,v}$, see Equation (3). Then, we apply recursively MΘRSE on all instances (S_i, ω) and (S_i, ω_i^*), with

$0 \leq i \leq \ell - 1$. Eventually, one can retrieve the eccentricites of (G, ω) in linear time according to Theorem 4.

We show that MΘRSE is executed in quasilinear time.

Theorem 1. *There exists a combinatorial algorithm which computes all weighted eccentricities of a weighted median graph (G, ω) in quasilinear time $O(n \log^4(n))$.*

Proof. Both Theorems 3 and 4 ensure us that MΘRSE computes exactly all eccentricities of the weighted median graph (G, ω). Our effort consists now in showing that its running time is quasilinear.

We prove the following statement: if $|V(G)| \geq 3$, the computation of MΘRSE(G, ω), in addition with a running time $O(|V(G)| \log^2 |V(G)|)$, launches recursive calls on a collection $\mathcal{C}$ of weighted subgraphs of G satisfying:

- $\sum_{(G', \omega') \in \mathcal{C}} |V(G')| \leq |V(G)|$
- $\max \{|V(G')| : (G', \omega') \in \mathcal{C}\} \leq |V(G)| \left(1 - \frac{1}{2 \log(|V(G)|)}\right)$

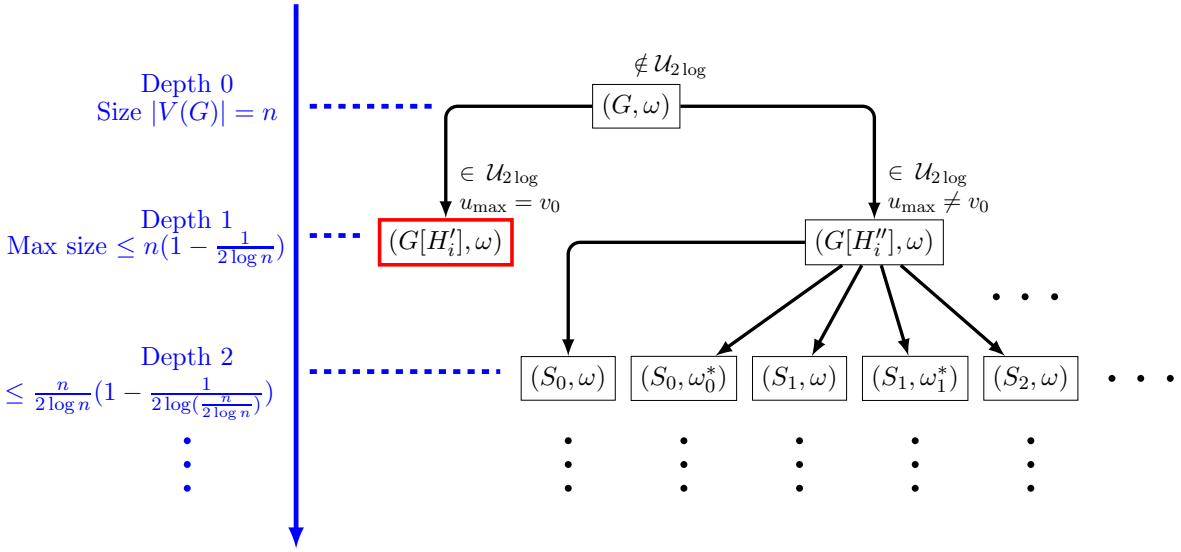

Figure 6: Tree of recursive calls: an example where the input (G, ω) admits a balanced Θ-class E_i; then $(G[H_i'], \omega) \in \mathcal{U}_{2\log}$ with $u_{\max} = v_0$ hence its weighted eccentricities can be directly computed in linear time (Corollary 2); finally $(G[H_i'''], \omega) \in \mathcal{U}_{2\log}$ but $u_{\max} \neq v_0$ so 2ℓ recursive calls are launched.

If (G, ω) admits a balanced Θ-class E_i, two recursive calls are achieved on its halfspaces which partition the vertex set of G. Moreover, from Definition 12, none of these halfspaces has a size greater than $n \left(1 - \frac{1}{2 \log n}\right)$ with $n = |V(G)|$. Otherwise, when $(G, \omega) \in \mathcal{U}_{2\log}$, the situation is trickier. If $u_{\max} = v_0$, all eccentricities can be computed in linear time (Corollary 2) and there is no recursive call (red node of the tree in Figure 6). But, if $u_{\max} \neq v_0$, two recursive calls are launched for each slice S_i, $0 \leq i \leq \ell - 1$: one with weight function ω and one with ω_i^*. Each slice is the subset of a minority halfspace, hence $|S_i| \leq \frac{n}{2 \log n} <$

$n\left(1-\frac{1}{2\log n}\right)$. Then, observe that two different slices do not intersect each other by definition. As $\ell \leq \log(|V(G)|)$, we have:

$$\sum_{(G',\omega')\in\mathcal{C}} |V(G')| = \sum_{i=0}^{\ell-1} 2|S_i| \leq \frac{2\ell|V(G)|}{2\log(|V(G)|)} \leq |V(G)|.$$

In both cases, the inequalities claimed above are satisfied. We analyze the tree of recursive calls, admitting the input (G,ω) as a root and, as leaves, either cases $u_{\max} = v_0$ or very small weighted graphs (at most two vertices). At each depth k of the tree, the total number of vertices involved in the instances is at most $n = |V(G)|$. The extra running time needed to retrieve the weighted eccentricities of the instances at depth k from the weighted eccentricities of instances at depth $k+1$ is $O(n\log^2 n)$: it is $O(n\log n)$ when a balanced Θ-class is present (Theorem 3), $O(n\log^2 n)$ otherwise (Corollary 3). Let s_k be the maximum number of vertices of some weighted graph at depth $k \geq 0$. Obviously, $s_0 = n$. Sequence (s_k) follows the scheme introduced in Lemma 16 with $\lambda = 2$: in particular, when $s_k \geq 3$, then $s_{k+1} \leq \lfloor s_k(1-\frac{1}{2\log(s_k)})\rfloor$. Indeed, according to the second inequality above, as s_k is the maximum size of some instance at depth k, any instance at depth $k+1$ cannot exceed $\lfloor s_k(1-\frac{1}{2\log(s_k)})\rfloor$. As a conclusion, the depth of the recursive tree cannot overpass $2\log^2 n$ (Lemma 16).

As the running time needed per depth is $O(n\log^2 n)$ and the depth of the tree of recursive calls is upper-bounded by $2\log^2 n$, then the total running time of MΘRSE is $O(n\log^4 n)$. □

5 Distance oracle

We present in this section the second outcome of this paper, which is the design of a distance oracle (DO) for (unweighted) median graphs with a poly-logarithmic size of labels and query time. In this section, the notions of balanced/unbalanced Θ-classes will be associated with a different function f than in Sections 3 and 4. Indeed, we fix f as the constant function: $f : n \to 3$. We denote by $\mathcal{U}_3$ the set of median graphs without any 3-balanced Θ-classes. In other words, if $G \in \mathcal{U}_3$, then for each $E_i \in \mathcal{E}(G)$, either $|H_i'| < \frac{n}{3}$ or $|H_i''| < \frac{n}{3}$.

The main technique used to achieve this goal is similar to the one used in Sections 3 and 4: we exploit balanced Θ-classes. We propose a recursive scheme where, at each step, a non-negligible number of vertices is withdrawn for each recursive call. When the number of vertices $n = |V(G)|$ is at most 2, the label of each vertex has size 0: it does not contain any bit. Indeed, the distance towards the other vertex is necessarily 1, as G is connected.

Now we fixed this trivial case, we focus on median graphs G with $n \geq 3$. We distinguish two cases: either G admits a 3-balanced Θ-class, or $G \in \mathcal{U}_3$.

5.1 Recursive scheme exploiting gated halfspaces

We propose a distance oracle Λ_G for median graphs G. Any label $\Lambda_G(u)$ is a sequence of bits. The size of the DO is the maximum size of all sequences $\Lambda_G(u)$: we denote $|\Lambda_G| = \max_{u\in V(G)} |\Lambda_G(u)|$. Then, we say that the DO Λ_G has a query time $\tau(n)$ if, for any pair $u, v \in V(G)$, the time needed to retrieve $d(u,v)$ thanks to labeling Λ_G is at most $\tau(n)$. The DO we propose has both poly-logarithmic size and query time.

Theorem 5. *Let G be a median graph and $E_i \in \mathcal{E}(G)$. Assume that:*

- *a DO $\Lambda_{H'_i}$ of graph $G[H'_i]$ is known with query time $\tau_1(|H'_i|)$,*
- *a DO $\Lambda_{H''_i}$ of graph $G[H''_i]$ is known with query time $\tau_2(|H''_i|)$.*

Then, one can build a DO Λ_G of graph G with size at most $3\log_2 n + 1 + \max\left\{|\Lambda_{H'_i}|, |\Lambda_{H''_i}|\right\}$ and query time $\tau(n) = O(\log_2 n) + \max\{\tau_1(|H'_i|), \tau_2(|H''_i|)\}$. The construction takes time $O(n(\log_2 n + \max\{|\Lambda_{H'_i}|, |\Lambda_{H''_i}|\}))$.

Proof. Thanks to our assumptions, every vertex of $V(G)$ is initially associated with a label of the halfspaces of E_i. The vertices $u' \in H'_i$ are labeled with $\Lambda_{H'_i}(u')$ while the vertices $u'' \in H''_i$ are labeled with $\Lambda_{H''_i}(u'')$.

For the construction of Λ_G, we begin with the computation of the gate of each vertex thanks to the BFS of Lemma 2. As a reminder, for any vertex $u' \in H'_i$, we determine its H''_i-gate $g(u') \in H''_i$, and conversely for each vertex $u'' \in H''_i$ we determine its H'_i-gate $g(u'')$. This operation is achieved in time $O(n \log n)$ by launching a BFS with a starting queue made up of H'_i, and a second one with a starting queue H''_i (as it was done with the eccentricities, see the proof of Theorem 3). At the end of the execution, for each $u \in V(G)$, we know all distances $d(u, g(u))$.

We are ready to describe the content of the DO Λ_G. For each $u \in V(G)$, the sequence of bits $\Lambda_G(u)$ contains successively:

1. $\Lambda_G^{\Theta}(u)$: the index i of the Θ-class E_i, encoded by $\log_2 n$ bits, since $q \leq n$,
2. $\Lambda_G^{\text{side}}(u)$: the side of u regarding E_i encoded with one bit: either $u \in H'_i$ or $u \in H''_i$,
3. $\Lambda_G^{\text{gate}}(u)$: the identity of its gate $g(u)$ through the opposite halfspace encoded with $\log_2 n$ bits,
4. $\Lambda_G^{\text{dist}}(u)$: the distance $d(u, g(u))$ encoded with $\log_2 n$ bits,
5. $\Lambda_G^{\text{rec}}(u)$: the label of u in its halfspace: if $u \in H'_i$, then we add $\Lambda_{H'_i}(u)$, otherwise $\Lambda_{H''_i}(u)$.

In summary, label $\Lambda_G(u)$ is the concatenation of $3\log_2 n + 1$ preliminary information in addition with the label of u in its E_i-halfspace: $\Lambda_G(u) = \Lambda_G^{\Theta}(u) \cdot \Lambda_G^{\text{side}}(u) \cdot \Lambda_G^{\text{gate}}(u) \cdot \Lambda_G^{\text{dist}}(u) \cdot \Lambda_G^{\text{rec}}(u)$. Hence, its size is at most $O(\log_2 n) + \max\{|\Lambda_{H'_i}|, |\Lambda_{H''_i}|\}$. Moreover, the time needed for the construction of Λ_G includes the BFS evoked in Lemma 2 but also the writing of sequence $\Lambda_G(u)$. The latter takes at most $O(\log_2 n + \max\{|\Lambda_{H'_i}|, |\Lambda_{H''_i}|\})$ and is executed for each vertex of G.

Let us explain how any distance $d(u, v)$ can be retrieved thanks to the DO Λ_G. From the $\log_2 n$ first bits of $\Lambda_G^{\Theta}(u)$ (or $\Lambda_G^{\Theta}(v)$), we obtain the Θ-class which was used as a separator. Then, we look at the next bit Λ_G^{side} for each vertex, it allows us to know into which halfspace u and v are. If they belong to the same halfspace , say H'_i w.l.o.g., then we obtain $d(u, v)$ thanks to the DO $\Lambda_{H'_i}$, given by both $\Lambda_G^{\text{rec}}(u)$ and $\Lambda_G^{\text{rec}}(v)$. Else, if $u \in H'_i$ and $v \in H''_i$ w.l.o.g., we first retrieve the gate $g(u)$ of u in H''_i thanks to the part $\Lambda_G^{\text{gate}}(u)$, together with the distance $d(u, g(u))$ thanks to $\Lambda_G^{\text{dist}}(u)$. Second, we determine the distance $d(g(u), v)$ with a query on the DO $\Lambda_{H''_i}$, given by both $\Lambda_G^{\text{rec}}(g(u))$ and $\Lambda_G^{\text{rec}}(v)$. Finally, we obtain $d(u, v) = d(u, g(u)) + d(g(u), v)$.

The query time consists in the analysis of the first $3\log_2 n + 1$ bits to retrieve the different information: E_i, halfspace of each vertex, gate of u and $d(u, g(u))$. Then, we necessarily launch

a query on either $\Lambda_{H_i'}$ or $\Lambda_{H_i''}$. All in all, the query time is upper-bounded by $O(\log_2 n) + \max\{\tau_1(|H_i'|), \tau_2(|H_i''|)\}$. □

We formulate an observation similar to the one we gave for the eccentricities problem after Theorem 3. We could apply recursively Theorem 5 until we find base cases, *i.e.* median graphs of at most 2 vertices. Unfortunately, if we select the Θ-class E_i arbitrarily, such an approach can lead to labels of linear size. However, in an utopian situation, we could select a balanced Θ-class E_i at each recursive step. In this way, the depth of the recursive tree would be logarithmic. The issue is that many median graphs do not admit any balanced Θ-class. For this reason, we provide in the next subsection a labeling procedure for graphs $G \in \mathcal{U}_3$.

5.2 Oracle construction for median graphs without balanced Θ-classes

In order to handle the case of median graphs without any balanced Θ-class, we label each vertex with not only its distance to the median vertex v_0, but also its gates for the "neighboring" fibers around it. In this way, one can retrieve any distance $d(u, v)$ by looking at the successive distances from gate to gate a logarithmic number of times. The query time is thus poly-logarithmic, such as the size of the labeling.

Consider a median graph $G \in \mathcal{U}_3$. According to Lemma 17, there is a unique median vertex v_0 for G belonging to all majority halfspaces. Each vertex $v \neq v_0$ admits a ladder set $L_{v_0,v}$, hence the vertices of G can be partitioned in function of their ladder $L_{v_0,v}$. We denote by $V_L \subseteq V(G) \setminus \{v_0\}$ the set of vertices v with ladder set $L_{v_0,v} = L$. Set V_L can also be seen, from Definition 10, as the intersection of all minority halfspaces of Θ-classes in L:

$$V_L = \bigcap_{E_i \in L} H_i'.$$

For any POF L containing only Θ-classes adjacent to v_0, V_L is not only nonempty but also gated, as shown in the next lemma.

Lemma 24. *For any POF L adjacent to v_0, V_L is nonempty and gated.*

Proof. As all Θ-classes of L are adjacent to v_0, there is a hypercube Q_L containing both v_0 and edges of L adjacent to v_0 (Lemma 10). We denote by v_L the farthest-to-v_0 vertex of Q_L, said differently the opposite vertex of v_0 in Q_L. By definition, the ladder set of v_L is L, therefore V_L is nonempty.

Let $\mathrm{St}(v_0)$ be the subgraph of G made up of the vertices which belong to a common induced hypercube with v_0. This notion was defined in [23] and called the *star* of a vertex. The authors (Proposition 2, [23]) proved that, for any vertex v, $\mathrm{St}(v)$ is a gated subgraph of G. Consequently, the fibers of $\mathrm{St}(v_0)$ are gated (Lemma 4).

We claim that V_L is exactly the fiber of vertex v_L: in brief, $V_L = F_{\mathrm{St}(v_0)}[v_L]$. Let $u \in V_L$ and $z \in \mathrm{St}(v_0)$: we show that $v_L \in I(u, z)$ necessarily. If $\sigma_{v_L,u} \cap \sigma_{v_L,z} = \emptyset$, then our claim holds, according to Lemma 9. By contradiction, assume that some Θ-class E_j belongs to both $\sigma_{v_L,u}$ and $\sigma_{v_L,z}$. As $E_j \in \sigma_{v_L,z}$, then by convexity of $\mathrm{St}(v_0)$, E_j is adjacent to v_0. However, at the same time, we have $L = L_{v_0,u} = \sigma_{v_0,v_L} \subseteq \sigma_{v_0,u}$. Hence, $v_L \in I(v_0, u)$ and $\sigma_{v_L,u} \subseteq \sigma_{v_0,u}$ does not contain any Θ-class adjacent to v_0 since all of them are present, by definition, in σ_{v_0,v_L}. So, E_j is not adjacent to v_0, which contradicts our first observation. As a consequence, v_L is the $\mathrm{St}(v_0)$-gate of any $u \in V_L$. As fibers are gated (Lemma 4), then we conclude that set V_L is gated. □

The idea beyond our DO follows. We do not give any label to the median vertex v_0. We exploit the partition $\{V_L : L \neq \emptyset,$ POF adjacent to $v_0\}$ of $V(G) \setminus \{v_0\}$. We compute recursively the DO of all median subgraphs V_L. Observe that, for any Θ-class $E_i \in L$, then $V_L \subseteq H_i'$ and hence each set contains at most $\frac{n}{3}$ vertices, since $G \in \mathcal{U}_3$. Then, we complete each label $\Lambda_{V_L}(u)$ with the identity of the ladder set $L = L_{v_0,u}$, the distance $d(v_0, u)$, and the pair gate-distance from u to its gate in any neighboring fiber $V_{L'}$ of V_L. The formal (recursive) definition of the DO Λ_G for $G \in \mathcal{U}_3$ is given below.

Definition 20 (DO Λ_G for $G \in \mathcal{U}_3$)**.** *Let $G \in \mathcal{U}_3$ with $n \geq 3$. The label $\Lambda_G(v_0)$ is empty. For $u \in V(G) \setminus \{v_0\}$, sequence $\Lambda_G(u)$ contains:*

1. *$\Lambda_G^L(u)$: the ladder set $L_{v_0,u}$, encoded with $(\log_2 n)^2$ bits,*
2. *$\Lambda_G^{\text{dist}}(u)$: the distance $d(v_0, u)$, encoded with $\log_2 n$ bits,*
3. *$\Lambda_G^{\text{fib}}(u)$: each triplet $(L \setminus L', g_{L'}(u), d(u, g_{L'}(u)))$, for all subsets $L' \subsetneq L = L_{v_0,u}$ with $|L'| = |L| - 1$, where $g_{L'}(u)$ is the gate of u for the gated fiber $V_{L'}$. Each triplet is encoded with $3 \log_2 n$ bits, so the sequence of all triplets has size $O((\log_2 n)^2)$,*
4. *$\Lambda_G^{\text{rec}}(u)$: the label $\Lambda_{V_L}(u)$ of the DO Λ_{V_L} computed recursively on the median subgraph $G[V_L]$.*

We have $\Lambda_G(u) = \Lambda_G^L(u) \cdot \Lambda_G^{\text{dist}}(u) \cdot \Lambda_G^{\text{fib}}(u) \cdot \Lambda_G^{\text{rec}}(u)$.

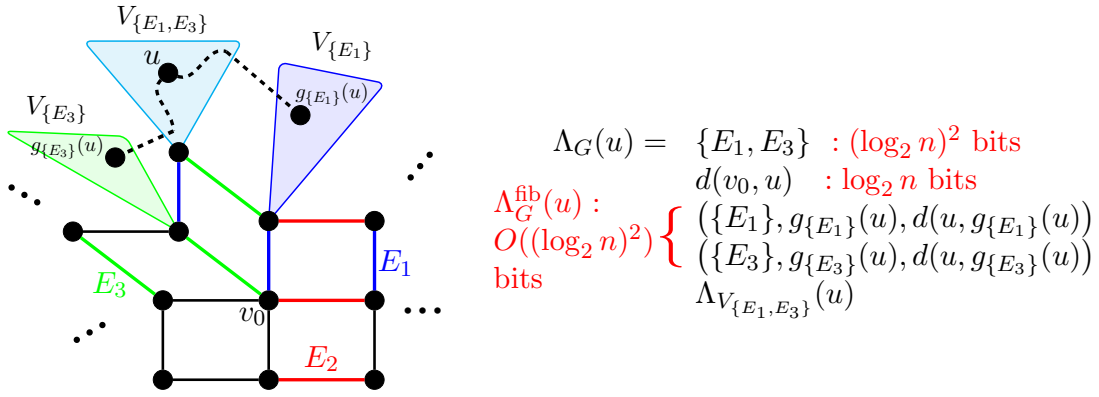

Figure 7: The label $\Lambda_G(u)$ of some vertex $u \in V(G)$, $G \in \mathcal{U}_3$

An example of label $\Lambda_G(u)$ is given in Figure 7. The definition of our labeling Λ_G is now complete. Remember that, if the graph G has at most 2 vertices, $|\Lambda_G| = 0$: there is no need to label the vertices since they are necessarily at distance 1. Else, we distinguish two cases. If G contains a 3-balanced Θ-class E_i, then we compute recursively the DO of each halfspace and label a logarithmic number of bits for each vertex in addition with the DO of its halfspace, as described in Theorem 5. If G contains no 3-balanced Θ-class, *i.e.* $G \in \mathcal{U}_3$, then we proceed as in Definition 20: we compute recursively the DO of each fiber V_L and label each vertex with a poly-logarithmic number of bits in addition with the DO of its fiber V_L. We thus conclude on the size of the labeling Λ_G.

Lemma 25. *For any median graph G, the size of Λ_G is poly-logarithmic: $|\Lambda_G| = O((\log_2 n)^3)$*

Proof. Let $\alpha(n)$ be the maximum size of a DO Λ_G on median graphs G satisfying $|V(G)| = n$. For $n \geq 3$, we show that $\alpha(n) \leq 4(\log_2 n)^2 + \log_2 n + \alpha(\lfloor \frac{2n}{3} \rfloor)$.

If G admits a 3-balanced Θ-class, as shown in Theorem 5, the size of the labeling Λ_G is at most $3\log_2 n + 1 + \alpha(\lfloor \frac{2n}{3} \rfloor)$ since each halfspace of a 3-balanced Θ-class contains at most $\lfloor \frac{2n}{3} \rfloor$ vertices.

However, if $G \in \mathcal{U}_3$, let us justify that, in this case, the size of the labeling Λ_G is upper-bounded by $4(\log_2 n)^2 + \log_2 n + \alpha(\lfloor \frac{n}{3} \rfloor)$. As already stated in Definition 20, the size of the first two parts of the labeling, *i.e.* Λ_G^L and Λ_G^{dist}, is $(\log_2 n)^2 + \log_2 n$. The third part Λ_G^{fib} is trickier to analyze. Let $u \in V(G)$ and $L = L_{v_0,u}$. The label $\Lambda_G^{\text{fib}}(u)$ contains as many triplets as the number of subsets L' of L with exactly one element left. As $|L| \leq d \leq \log_2 n$, there are at most $\log_2 n$ such subsets. The triplet is made up of the singleton $L \setminus L'$, the gate $g_{L'}(u)$ and the distance $d(u, g_{L'}(u))$. Each component of the triplet can be encoded with $\log_2 n$ bits. Therefore, the size of $\Lambda_G^{\text{fib}}(u)$ is at most $3(\log_2 n)^2$. Finally, $\Lambda_G(u)$ also contains the part $\Lambda_G^{\text{rec}}(u)$, which is the label of u for the DO on graph $G[V_L]$. As $|V_L| < \frac{n}{3}$, we have $|\Lambda_G(u)| \leq 4(\log_2 n)^2 + \log_2 n + \alpha(\lfloor \frac{n}{3} \rfloor)$.

Combining the inequalities obtained for the two possible cases, *i.e.* $G \in \mathcal{U}_3$ or $G \notin \mathcal{U}_3$, we have: $\alpha(n) \leq 4(\log_2 n)^2 + \log_2 n + \alpha(\lfloor \frac{2n}{3} \rfloor)$. Furthermore, as a base case, $\alpha(1) = \alpha(2) = 0$. The depth of this recursive sequence is logarithmic since we divide at each recursive step the number of vertices of the graph by a constant factor smaller than 1. The extra size added into the labeling at each recursive step is $O((\log_2 n)^2)$. We conclude: $\alpha(n) = O((\log_2 n)^3)$. □

The size of labeling Λ_G for any median graph G is guaranteed to be poly-logarithmic. We show now that obtaining labels $\Lambda_G(u)$ for each vertex $u \in V(G)$ can be achieved in quasilinear time.

Lemma 26. *There is a combinatorial algorithm which, given any median graph G, outputs the labeling Λ_G in quasilinear time $O(n(\log_2 n)^4)$.*

Proof. Let $T(n)$ be the maximum construction time of a labeling Λ_G for median graphs G satisfying $|V(G)| = n$. We almost entierely handled the case where $G \notin \mathcal{U}_3$: in Theorem 5, we proved that, given a balanced Θ-class E_i and the labeling $\Lambda_{H_i'}$ and $\Lambda_{H_i''}$ of its halfspaces, we build the labeling Λ_G in time $O(n(\log_2 n + \max\{|\Lambda_{H_i'}|, |\Lambda_{H_i''}|\}))$. Now, we naturally add the construction time for labelings $\Lambda_{H_i'}$ and $\Lambda_{H_i''}$ which were supposed to be known in Theorem 5. Moreover, thanks to Lemma 25, values $|\Lambda_{H_i'}|$ and $|\Lambda_{H_i''}|$ can be upper-bounded by $O((\log_2 n)^3)$. In summary, we obtain: $T(n) \leq O(n(\log_2 n)^3) + T(\lambda n) + T(\mu n)$, where $\lambda = \frac{|H_i'|}{n}$ and $\mu = \frac{|H_i''|}{n}$, hence $\lambda + \mu = 1$ and $\max\{\lambda, \mu\} \leq \frac{2}{3}$.

We focus now on the case $G \in \mathcal{U}_3$. Let $u \in V(G) \setminus \{v_0\}$. First, a BFS starting from v_0 gives us the ladder $L_{v_0,u}$ (Lemma 13) together with the distance $d(v_0, u)$. Second, we present a procedure to obtain all labels $\Lambda_G^{\text{fib}}(u)$. We begin with looking at all edges of the graph: if they connect a vertex of a fiber V_L with another vertex of a fiber $V_{L'}$, where $L' \subsetneq L$, $|L'| = |L| - 1$, we associate this edge to the pair (L, L'). We denote by $\partial E_{L,L'}$ this kind of edges and by $\partial V_{L,L'}$ the set of vertices of $V_{L'}$ which admit a neighbor in V_L.

For any POF L adjacent to v_0 and any subset $L' \subsetneq L$, $|L'| = |L| - 1$, we launch a BFS on graph $G_{L,L'} = (V_L \cup \partial V_{L,L'}, E[V_L] \cup \partial E_{L,L'})$ in order to determine, for each $u \in V_L$, its $V_{L'}$-gate

$g_{L'}(u)$ and also $d(u, g_{L'}(u))$ (Lemma 2). The running time of this operation depends on the total number of edges of all graphs $G_{L,L'}$. Any edge of V_L belongs to at most $d \leq \log_2 n$ such graphs, as there are $|L|$ possible subsets L' of L. The edges connecting two sets V_L and $V_{L'}$ are thus taken into account in only one graph $G_{L,L'}$. All in all, as each edge is considered in at most $\log_2 n$ such graphs $G_{L,L'}$, the total running time of this procedure is $O(m \log_2 n) = O(n(\log_2 n)^2)$. With the writing of each label, the total time spent to determine blocks Λ_G^L, Λ_G^{dist} and Λ_G^{fib} is $O(n(\log_2 n)^3)$. Finally, we need to compute recursively the DO of each fiber V_L. In brief, $T(n) \leq O(n(\log_2 n)^3) + \sum_L T(|V_L|)$. As each $|V_L| < \frac{n}{3}$, we can write: $T(n) \leq O(n(\log_2 n)^3) + \sum_k T(\alpha_k n)$, where $\sum_k \alpha_k = 1$ and $\max\{\alpha_k\} < \frac{1}{3}$.

Both inequations for $T(n)$ fall into the domain of the Master theorem described in [24]. At each recursive step, an extra time $O(n(\log_2 n)^3)$ is needed and, at the same time, the size of the instances on which recursive calls are launched is decreased by a constant factor. Moreover, the total number of vertices considered at each depth of the recursive tree stays equal to n. All in all, we have: $T(n) = O(n(\log_2 n)^4)$. □

The last part of our proof is certainly the most important of course: even if the labeling Λ_G has poly-logarithmic size and can be computed in quasilinear time, we do not know yet whether it allows us to retrieve any distance in poly-logarithmic time. The algorithm we present below use a new technique for the case $G \in \mathcal{U}_3$: to retrieve distance $d(u, v)$, we compare the ladder sets of both vertices, *i.e.* $L_{v_0,u}$ and $L_{v_0,v}$. The idea consists in going from u to v via "neighboring" ladder sets (differing from only one Θ-class), using the information of labeling Λ_G^{fib}.

Theorem 6. *Let G be a median graph and $u, v \in V(G)$. One can retrieve any distance $d(u, v)$ thanks to Λ_G with poly-logarithmic query time $O(\log^4(n))$.*

Proof. For graphs with at most two vertices, one can retrieve distances in constant time. We proceed by induction on the size of $|V(G)|$ to prove that labeling Λ_G is a DO. For median graphs $G \notin \mathcal{U}_3$: according to Theorem 5, if the DOs $\Lambda_{H_i'}$ and $\Lambda_{H_i''}$ are known, then the labeling Λ_G we built is a DO. By induction hypothesis, our labeling is a DO for graphs with a smaller number of vertices, in particular $G[H_i']$ and $G[H_i'']$. So, Λ_G is a DO for $G \notin \mathcal{U}_3$ with query time $\tau(n) = O(\log_2 n) + \max\{\tau(|H_i'|), \tau(|H_i''|)\} \leq O(\log_2 n) + \tau(\lfloor \frac{2n}{3} \rfloor)$.

Let us focus on the more complicated case $G \in \mathcal{U}_3$: our objective is to retrieve a distance $d(u, v)$, for some pair $u, v \in V(G)$. A first simple case occurs when one of these vertices is v_0, w.l.o.g. $u = v_0$. The label $\Lambda_G^{\text{dist}}(v)$, which is a part of $\Lambda_G(v)$, gives directly distance $d(v_0, v)$. Hence, we retrieve this distance in constant time.

Assume that both u and v are different from v_0. Both admit a ladder set with v_0, which can be deduced from $\Lambda_G^L(u)$ and $\Lambda_G^L(v)$. Say, w.l.o.g., that $|L_{v_0,u}| \leq |L_{v_0,v}|$. Another simple case is when $L_{v_0,u} = L_{v_0,v}$. The distance $d(u, v)$ can be computed thanks to a query on the labels $\Lambda_{V_L}(u)$ and $\Lambda_{V_L}(v)$, where $L = L_{v_0,u}$. Indeed, V_L is gated, so $G[V_L]$ is a median subgraph of G. These two labels are given by the part Λ_G^{rec} of Λ_G over each vertex. The query time in this situation is thus $\tau(\lfloor \frac{n}{3} \rfloor)$ by induction hypothesis, as $|V_L| < \frac{n}{3}$.

Nevertheless, in general, $L_{v_0,u} \neq L_{v_0,v}$. A third case easy to handle is when $L_{v_0,u} \cap L_{v_0,v} = \emptyset$. In this scenario, the signatures $\sigma_{v_0,u}$ and $\sigma_{v_0,v}$ have no Θ-class in common: if they had one, say E_i, by convexity of its boundary $\partial H_i''$, then $v_0 \in \partial H_i''$ and E_i must belong to both ladder sets from Definition 10. So, $v_0 \in I(u, v)$ and the distance $d(u, v) = d(u, v_0) + d(v_0, v)$ can be retrieved from labels $\Lambda_G^{\text{dist}}(u)$ and $\Lambda_G^{\text{dist}}(v)$.

The remaining (and hard) case is the following one: $L_{v_0,u} \neq L_{v_0,v}$ and $L_{v_0,u} \cap L_{v_0,v} \neq \emptyset$. First, we compute the *ladder sequence* $\mathcal{S}_{u,v}$ between these two sets $L_{v_0,u}$ and $L_{v_0,v}$: this a finite sequence of POFs, starting from $L_{v_0,u}$ and finishing at $L_{v_0,v}$. The size gap between two consecutive POFs of this sequence is exactly 1. The first part of $\mathcal{S}_{u,v}$ starts with $L_{v_0,u}$ and goes towards $L_{v_0,u} \cap L_{v_0,v}$. At each step, a Θ-class of $L_{v_0,u} \setminus L_{v_0,v}$ is withdrawn from the current POF. The second part of $\mathcal{S}_{u,v}$ starts with $L_{v_0,u} \cap L_{v_0,v}$ and goes towards $L_{v_0,v}$ by adding to the current POF a Θ-class of $L_{v_0,v} \setminus L_{v_0,u}$ at each step. The elements of sequence $\mathcal{S}_{u,v}$ are denoted by:

$$\mathcal{S}_{u,v} = \left(L^{(-r)}, L^{(-r+1)}, \ldots, L^{(-1)}, L^{(0)}, L^{(1)}, \ldots, L^{(t-1)}, L^{(t)}\right),$$

where r and t are nonnegative integers, $L^{(-r)} = L_{v_0,u}$, $L^{(t)} = L_{v_0,v}$, and $L^{(0)} = L_{v_0,u} \cap L_{v_0,v}$. For any integer $0 \leq i \leq r-1$, $L^{(-i+1)}$ is obtained from $L^{(-i)}$ by withdrawing one Θ-class belonging to $L^{(-i)} \setminus L_{v_0,v}$. Similarly, for any $0 \leq j \leq t-1$, $L^{(j)}$ is obtained from $L^{(j+1)}$ by withdrawing one Θ-class of $L^{(j+1)} \setminus L_{v_0,u}$. As an example, if $L_{v_0,u} = \{E_1, E_2, E_3\}$ and $L_{v_0,v} = \{E_3, E_4, E_5\}$, then $\mathcal{S}_{u,v}$ might be: $(\{E_1, E_2, E_3\}, \{E_2, E_3\}, \{E_3\}, \{E_3, E_4\}, \{E_3, E_4, E_5\})$. As each ladder set has size at most d, the length of sequence $\mathcal{S}_{u,v}$ is upper-bounded by $2d$.

For any integer $0 \leq i \leq r-1$, we denote by $E^{(-i)}$ the Θ-class of the singleton $L^{(-(i+1))} \setminus L^{(-i)}$. Similarly, $E_{(j)}$ denotes the Θ-class of $L^{(j+1)} \setminus L^{(j)}$, for $0 \leq j \leq t-1$. We show that there is a shortest (u,v)-path passing through all sets V_L for $L \in \mathcal{S}_{u,v}$. We proceed in three steps, each one characterized by a claim.

Claim 1: the median of triplet u, v, v_0, vertex $m = m(u, v, v_0)$ (see Definition 4), belongs to $V_{L^{(0)}}$. As $m \in I(u,v)$, then it belongs, by convexity, to the minority halfspaces of all Θ-classes in $L_{v_0,u} \cap L_{v_0,v}$. But, as $m \in I(v, v_0)$ and both v and v_0 are into the majority halfspace of all Θ-classes of $L_{v_0,u} \setminus L_{v_0,v}$, then m is also in the majority halfspace of these Θ-classes. Observe that the same argument holds for $L_{v_0,v} \setminus L_{v_0,u}$, because $m \in I(u, v_0)$. In summary, m belongs to the majority halfspace of all Θ-classes in the symmetric difference between $L_{v_0,u}$ and $L_{v_0,v}$, but to the minority halfspace of all Θ-classes in $L_{v_0,u} \cap L_{v_0,v}$. Therefore, $L_{v_0,m} = L_{v_0,u} \cap L_{v_0,v}$.

Claim 2: there exists a shortest (u,m)-path passing through all fibers $V_{L^{(-i)}}$, $0 \leq i \leq r$. We show briefly that a way to go from u to m is to traverse fiber $V_{L^{(-r+1)}}$. We will see that, applying the same argument iteratively, leads to the same conclusion than Claim 2. Vertex u belongs to the minority halfspace of $E^{(-r+1)}$ while m belongs to its majority halfspace. So, u admits a gate $g_{-r+1}(u)$ for the majority halfspace of $E^{(-r+1)}$. Vertex $g_{-r+1}(u)$ belongs to $V_{L^{(-r+1)}}$. Then, we can pursue with the same argument for the gate $g_{-r+1}(u)$: it admits a gate $g_{-r+2}(u)$ in the majority halfspace of $E^{(-r+2)}$, and so on. Finally, we produce a sequence of gates, all belonging by definition to $I(u,m)$. The last gate of this sequence, $g_0(u)$, belongs to $V_{L^{(0)}}$, such as m, hence the last part of the shortest (u,m)-path stays in $V_{L^{(0)}}$, by convexity of fibers.

Claim 3: there exists a shortest (v,m)-path passing through all fibers $V_{L^{(j)}}$, $0 \leq j \leq t$. The argument is the same as the one for Claim 2: we start from v and determine iteratively the successive gates on the majority halfspaces of respectively Θ-classes $E^{(t-1)}, E^{(t-2)}, \ldots$

As a conclusion of the three claims, since $m \in I(u,v)$ belongs to $V_{L^{(0)}}$, we ensure that there exists a shortest (u,v)-path, consisting first in a section from u to m passing through all fibers indexed negatively (Claim 2), second in a section from m to v passing through all fibers indexed positively (Claim 3). In brief, there is a shortest (u,v)-path passing successively through the fibers V_L, where L follows the sequence $\mathcal{S}_{u,v}$. An example is provided in Figure 8,

with a sequence $\mathcal{S}_{u,v} = (\{E_1, E_3\}, \{E_1\}, \{E_1, E_2\})$. The orange dashed path represents a shortest (u, v)-path traversing all successive fiber gates.

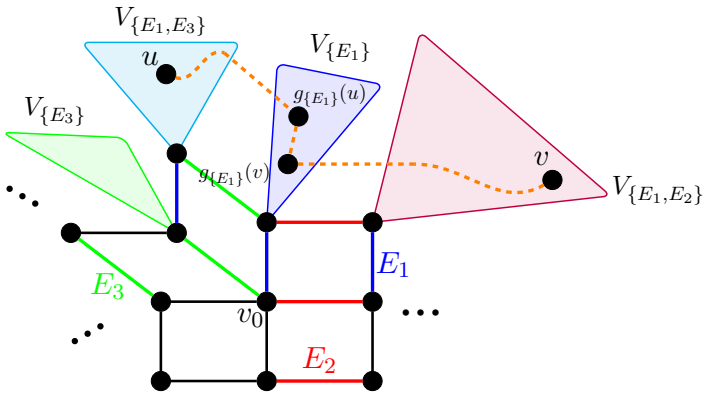

Figure 8: A concrete view of a median graph $G \in \mathcal{U}_3$ and its fiber to show how we retrieve distance $d(u, v)$, with $L_{v_0,u} = \{E_1, E_3\}$ and $L_{v_0,v} = \{E_1, E_2\}$

In order to retrieve distance $d(u, v)$, we use some labels of each fiber V_L, where $L \in \mathcal{S}_{u,v}$. From label $\Lambda_G^{\text{fib}}(u)$, we obtain the gate of u for the set $V_{L^{(-r+1)}}$, which is necessarily $g_{-r+1}(u)$ since $V_{L^{(-r+1)}}$ is a subset of the majority halfspace of $E^{(-r+1)}$. Then, from $\Lambda_G^{\text{fib}}(g_{-r+1}(u))$, we obtain the second gate, which belongs to set $V_{L^{(-r+2)}}$, and so on. In summary, we follow the sequence $\mathcal{S}_{u,v}$ of logarithmic size and obtain from Λ_G^{fib} successively the pair gate/distance in the next majority halfspace considered. By simply summing up all the distances between the different gates, we obtain $d(u, v)$. The distance that remains unknown even after the pick up of pairs gate/distance is the distance between the two gates obtain in fiber $V_{L^{(0)}}$ (*e.g.* vertices $g_{\{E_1\}}(u)$ and $g_{\{E_1\}}(v)$ in Figure 8). To compute the distance between these two final gates, a query on Λ_G^{rec} for these gates suffices. Hence, distance $d(u, v)$ is now determined.

As u and v belong to different fibers but with a nonempty intersection, the whole process consists in (i) retrieving the ladder set of both vertices thanks to Λ_G^L, (ii) computing the sequence $\mathcal{S}_{u,v}$ and then the successive gates thanks to Λ_G^{fib}, (iii) obtaining the distance between all successive gates thanks to Λ_G^{fib}, and (iv) computing the distance between the gates of $V_{L^{(0)}}$ thanks to Λ_G^{rec}. The cost of (i) is negligible compared to the one of (ii) and (iii) together, which is $O((\log_2 n)^3)$, since the computation of one gate takes $O(|\Lambda_G^{\text{fib}}|) = O((\log_2 n)^2)$ and there are at most $2d$ of them. So, taking also (iv) into account, we have $\tau(n) = \tau(\lfloor \frac{n}{3} \rfloor) + O((\log_2 n)^3)$.

Considering all possible cases, an upper bound is $\tau(n) = \tau(\lfloor \frac{2n}{3} \rfloor) + O((\log_2 n)^3)$, consequently, by Master theorem, $\tau(n) = O((\log_2 n)^4)$. □

The labeling Λ_G thus guarantees all the properties listed in the following statement, which is the second contribution of our paper.

Theorem 2. *There exists a combinatorial algorithm which computes in quasilinear time $O(n \log^4(n))$, for any median graph G, vertex-labels $(\Lambda_G(u))_{u \in V(G)}$ of size $O(\log^3(n))$ such that the distance $d(x, y)$ between a pair x, y of vertices can be retrieved in time $O(\log^4(n))$ thanks to the labels.*

References

[1] A. Abboud, F. Grandoni, and V. V. Williams. Subcubic equivalences between graph centrality problems, APSP and diameter. In *Proc. of SODA*, pages 1681–1697, 2015.

[2] H. Bandelt. Retracts of hypercubes. *Journal of Graph Theory*, 8(4):501–510, 1984.

[3] H. Bandelt and J. Barthélemy. Medians in median graphs. *Discret. Appl. Math.*, 8(2):131–142, 1984.

[4] H. Bandelt and V. Chepoi. Metric graph theory and geometry: a survey. *Contemp. Math.*, 453:49–86, 2008.

[5] H. Bandelt, V. Chepoi, A. W. M. Dress, and J. H. Koolen. Combinatorics of lopsided sets. *Eur. J. Comb.*, 27(5):669–689, 2006.

[6] H. Bandelt, V. Chepoi, and D. Eppstein. Combinatorics and geometry of finite and infinite squaregraphs. *SIAM J. Discret. Math.*, 24(4):1399–1440, 2010.

[7] H. Bandelt, P. Forster, and A. Röhl. Median-joining networks for inferring intraspecific phylogenies. *Molecular Biology and Evolution*, 16(1):37–48, 1999.

[8] H. Bandelt, P. Forster, B. C. Sykes, and M. B. Richards. Mitochondrial portraits of human populations using median networks. *Genetics*, 141(2):743–753, 1995.

[9] H. Bandelt, V. Macaulay, and M. Richards. Median networks: Speedy construction and greedy reduction, one simulation, and two case studies from human mtDNA. *Molecular Phylogenetics and Evolution*, 16(1):8–28, 2000.

[10] H. Bandelt and H. M. Mulder. Pseudo-median graphs: decomposition via amalgamation and cartesian multiplication. *Discret. Math.*, 94(3):161–180, 1991.

[11] H. Bandelt, L. Quintana-Murci, A. Salas, and V. Macaulay. The fingerprint of phantom mutations in mitochondrial DNA data. *Am. J. Hum. Genet.*, 71:1150–1160, 2002.

[12] J. Barthélemy and J. Constantin. Median graphs, parallelism and posets. *Discret. Math.*, 111(1-3):49–63, 1993.

[13] B. Ben-Moshe, B. K. Bhattacharya, Q. Shi, and A. Tamir. Efficient algorithms for center problems in cactus networks. *Theor. Comput. Sci.*, 378(3):237–252, 2007.

[14] L. Bénéteau, J. Chalopin, V. Chepoi, and Y. Vaxès. Medians in median graphs and their cube complexes in linear time. In *Proc. of ICALP*, volume 168, pages 10:1–10:17, 2020.

[15] P. Bergé, G. Ducoffe, and M. Habib. Subquadratic-time algorithm for the diameter and all eccentricities on median graphs. In *Procs. of STACS*, volume 219 of *LIPIcs*, pages 9:1–9:21, 2022.

[16] P. Bergé and M. Habib. Diameter, radius and all eccentricities in linear time for constant-dimension median graphs. In *Proc. of LAGOS*, 2021.

[17] B. Brešar. Characterizing almost-median graphs. *Eur. J. Comb.*, 28(3):916–920, 2007.

[18] B. Brešar, S. Klavžar, and R. Skrekovski. On cube-free median graphs. *Discret. Math.*, 307(3-5):345–351, 2007.

[19] M. Chastand. Fiber-complemented graphs – I: structure and invariant subgraphs. *Discret. Math.*, 226(1-3):107–141, 2001.

[20] C. T. Cheng. A poset-based approach to embedding median graphs in hypercubes and lattices. *Order*, 29(1):147–163, 2012.

[21] V. Chepoi. Graphs of some CAT(0) complexes. *Adv. Appl. Math.*, 24(2):125–179, 2000.

[22] V. Chepoi, A. Labourel, and S. Ratel. Distance and routing labeling schemes for cube-free median graphs. *CoRR*, abs/1809.10508, 2018.

[23] V. Chepoi, A. Labourel, and S. Ratel. Distance labeling schemes for cube-free median graphs. In *Proc. of MFCS*, volume 138, pages 15:1–15:14, 2019.

[24] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms, 3rd Edition*. MIT Press, 2009.

[25] D. Djoković. Distance-preserving subgraphs of hypercubes. *Journal of Combinatorial Theory, Series B*, 14(3):263–267, 1973.

[26] A. Gély, M. Couceiro, L. Miclet, and A. Napoli. A study of algorithms relating distributive lattices, median graphs, and Formal Concept Analysis. *Int. J. Approx. Reason.*, 142:370–382, 2022.

[27] M. Gromov. *Hyperbolic Groups*, pages 75–263. Springer New York, 1987.

[28] J. Hagauer, W. Imrich, and S. Klavžar. Recognizing median graphs in subquadratic time. *Theor. Comput. Sci.*, 215(1-2):123–136, 1999.

[29] R. Hammack, W. Imrich, and S. Klavžar. *Handbook of Product Graphs, Second Edition*. CRC Press, Inc., 2011.

[30] M. Kovše. Complexity of phylogenetic networks: counting cubes in median graphs and related problems. *Analysis of complex networks: From Biology to Linguistics*, pages 323–350, 2009.

[31] R. J. Lipton and R. E. Tarjan. A separator theorem for planar graphs. *SIAM Journal on Applied Mathematics*, 36(2):177–189, 1979.

[32] F. R. McMorris, H. M. Mulder, and F. S. Roberts. The median procedure on median graphs. *Discret. Appl. Math.*, 84(1-3):165–181, 1998.

[33] H. M. Mulder and A. Schrijver. Median graphs and Helly hypergraphs. *Discret. Math.*, 25(1):41–50, 1979.

[34] M. Mulder. The structure of median graphs. *Discret. Math.*, 24(2):197–204, 1978.

[35] M. Mulder. The interval function of a graph. *Mathematical Centre Tracts, Mathematisch Centrum, Amsterdam*, 1980.

[36] D. H. Parks and R. G. Beiko. Measuring Community Similarity with Phylogenetic Networks. *Molecular Biology and Evolution*, 29(12):3947–3958, 2012.

[37] V. Sassone, M. Nielsen, and G. Winskel. A classification of models for concurrency. In *Proc. of CONCUR*, volume 715 of *Lecture Notes in Computer Science*, pages 82–96, 1993.

[38] T. J. Schaefer. The complexity of satisfiability problems. In *Procs. of STOC*, pages 216–226, 1978.

[39] A. Vesel. Recognizing pseudo-median graphs. *Discret. Appl. Math.*, 116(3):261–269, 2002.

[40] P. M. Winkler. Isometric embedding in products of complete graphs. *Discret. Appl. Math.*, 7(2):221–225, 1984.

[41] B. Zimmermann, A. Röck, G. Huber, T. Krämer, P. M. Schneider, and W. Parson. Application of a west eurasian-specific filter for quasi-median network analysis: Sharpening the blade for mtDNA error detection. *Forensic Science International: Genetics*, 5(2):133–137, 2011.