Deletion only Dynamic Connectivity for Disk Graphs*

Haim Kaplan[1], Katharina Klost[2], Kristin Knorr†[2], Wolfgang Mulzer‡[2], and Liam Roditty[3]

1 **School of Computer Science, Tel Aviv University**
haimk@tau.ac.il
2 **Institut für Informatik, Freie Universität Berlin**
{kathklost, knorrkri, mulzer}@inf.fu-berlin.de
3 **Department of Computer Science, Bar Ilan University**
liamr@macs.biu.ac.il

Abstract

Let $S \subset \mathbb{R}^2$ be a set of n sites where $s \in S$ has an associated radius r_s defining a disk D_s. Then the *disk graph* $D(S)$ has a vertex for every site and two sites s, t are connected by an undirected edge if and only if $D_s \cap D_t \neq \emptyset$. We present a data structure which serves a sequence of n deletions and k connectivity queries in time $O(n \operatorname{polylog} n + k(\log n / \log \log n))$.

1 Introduction

The question if two vertices in a given graph are connected is crucial for many graph algorithms. The graph might be stored in a specialized data structure if multiple queries are given. If the graph is static, using BFS and labeling the vertices gives an optimal data structure. However if edge insertions or deletions are allowed things get more complicated. If both insertions and deletions are allowed, the data structure of Holm et al. [4] which can handle edge updates in time $O(n \log^2 n)$ and queries in amortized time $O(\frac{\log n}{\log \log n})$ is the best currently known data structure for general graphs.

We consider the problem of constructing a deletion-only dynamic connectivity data structure for disk graphs. Given a set $S \subseteq \mathbb{R}^2$ of n sites with associated radii r_s, the disk graph is the intersection graph of the disks D_s induced by the sites and their radii. While the description of $D(S)$ has size $O(n)$, it might have $\Theta(n^2)$ edges. So when starting with some disk graph and subsequently deleting sites, over time up to $\Theta(n^2)$ edges are deleted. We describe a data structure whose overall running time for the sequence of n site deletion is $o(n^2)$. For unit disk graphs, such a data structure is already known [6].

On a high level, our approach works as follows. We define a sparse proxy graph $H = (V, E)$ with $S \subseteq V$ which perfectly represents the connectivity in $D(S)$. We then store H in the data structure of Holm et al. and describe how to update H on the deletion of a site.

2 Preliminaries

Our data structures relies on combining various data structures and techniques. We briefly recall their definition and relevant properties here.

* Supported in part by grant 1367/2016 from the German-Israeli Science Foundation (GIF).
† Supported by the German Science Foundation within the research training group 'Facets of Complexity' (GRK 2434).
‡ Supported in part by ERC StG 757609.

37th European Workshop on Computational Geometry, St. Petersburg, Russia, April 7–9, 2021.
This is an extended abstract of a presentation given at EuroCG'21. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

Without loss of generality we scale and translate S such that the minimum distance between two sites is $1+\varepsilon$ and that the point set fits into a square with diameter $2^{\lceil \log \operatorname{diam}(S) \rceil}$. Furthermore we set all radii to $\min\{r_s, \operatorname{diam}(S)\}$. The *quadtree* on S is now defined in the usual fashion as a tree of degree at most 4 with the square with diameter $2^{\lceil \log \operatorname{diam}(S) \rceil}$ being the root. We will not explicitly distinguish between the cells and the vertices of the quadtree. Each cell of diameter 2^i which contains at least two sites is subdivided into up to four non-empty cells of diameter 2^{i-1}. By the assumptions on our point set, no cell of diameter 1 is subdivided further. As the height of the quadtree defined this way is $O(\log \operatorname{diam}(S))$, and thus does not depend on n, we consider the *compressed quadtree*. Let $\sigma_1, \ldots, \sigma_k$ be maximal path of vertices with degree one in the quadtree. In the compressed quadtree $\mathcal{Q}$ this path is replaced by the edge $\sigma_1\sigma_k$. It is well known that such a compressed quadtree has $O(n)$ vertices, height $O(n)$ and can be constructed in $O(n \log n)$ time [1, 3]. Given a cell $\sigma \in \mathcal{Q}$ we denote by $|\sigma|$ its diameter.

We will use a heavy path decomposition on $\mathcal{Q}$. Slightly adapting the terminology of Sleator and Tarjan [7], we call an edge uv of a tree *heavy*, if v is the first child of u that maximized the total number of nodes in the subtree rooted at v and *light* otherwise. A *heavy path* is a maximal path which consists of heavy edges. The set of all heavy paths is called the *heavy path decomposition*.

▶ **Lemma 2.1** (Sleator and Tarjan [7]). *Let T be a tree with n vertices. Then, the following properties hold:* ***1.*** *Every leaf-root path in T contains $O(\log n)$ light edges;* ***2.*** *every vertex of T lies on exactly one heavy path; and* ***3.*** *the heavy path decomposition of T can be constructed in $O(n)$ time.*

Let X be a linearly ordered set. We aim to find a set $\mathcal{X}$ of subsets of X, such that $|\mathcal{X}| = O(|X|)$ and every contiguous subsequence can be partitioned into $O(\log |X|)$ subsets from $\mathcal{X}$. Using a standard approach [2, 8] this can be achieved by building a perfect binary search tree with the elements of X in the leaves. The interval can now be represented by $O(\log n)$ subtrees along the search paths for its boundaries. Finally we will use the following data structure:

▶ **Lemma 2.2** (Reveal data structure (RDS), Kaplan et al. [5]). *Let R and B be site sets with $|R| + |B| = n$. There is a data structure which after preprocessing allows to delete sites from R and B. After deleting a site from R it reports all disks from B now disconnected from $\bigcup_{s \in R} D_s$. Preprocessing, deleting m sites from R and an arbitrary number of sites from B takes $O((n \log^5 n + m \log^9 n)\lambda_6(\log n))$ expected time and $O(n \log^3 n)$ expected space, where $\lambda_s(n)$ is the maximum length of a Davenport-Schinzel sequence.*

3 The Proxy Graph

In order to describe H we first augment the compressed quadtree $\mathcal{Q}$ on S by adding additional cells. Given a site $t \in S$, let σ be the cell with $t \in \sigma$ and $|\sigma| \leq r_t \leq 2|\sigma|$. Let $N(t)$ be the 13×13 neighborhood of σ. Observe that all sites s with $r_s \leq r_t$ which can form an edge with r_t lie in a disk of radius at most $2r_t \leq 4|\sigma|$. All cells intersecting or containing that disk are part of $N(t)$, see Figure 1. We augment $\mathcal{Q}$ by adding the neighborhood $N(t)$ of all sites and call the resulting tree $\mathcal{Q}'$.

Considering the heavy path decomposition $\mathcal{R}$ of $\mathcal{Q}'$, we further decompose each heavy path $R \in \mathcal{R}$ into *canonical paths* using the technique from section 2. The set $\mathcal{P}$ of all canonical paths received this way has the following properties, which follow from those of $\mathcal{R}$ and $\mathcal{P}$. See Figure 2 for an illustration of the lemma.

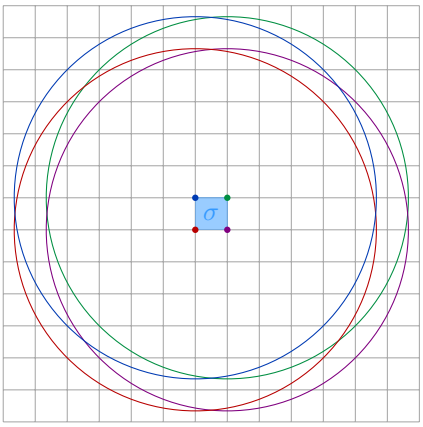

Figure 1 The disks with center in σ and radius $4|\sigma|$ are contained in $N_{13\times 13}(\sigma)$.

▶ **Lemma 3.1.** *Let σ be a cell of $\mathcal{Q}'$ and let π be the path in $\mathcal{Q}'$ from σ to the root. There exists a set $\mathcal{P}_\pi$ of canonical paths such that the following holds:* ***1.*** *$|\mathcal{P}_\pi| = O(\log^2 n)$,* ***2.*** *$\pi$ is the disjoint union of the canonical paths in $\mathcal{P}_\pi$*

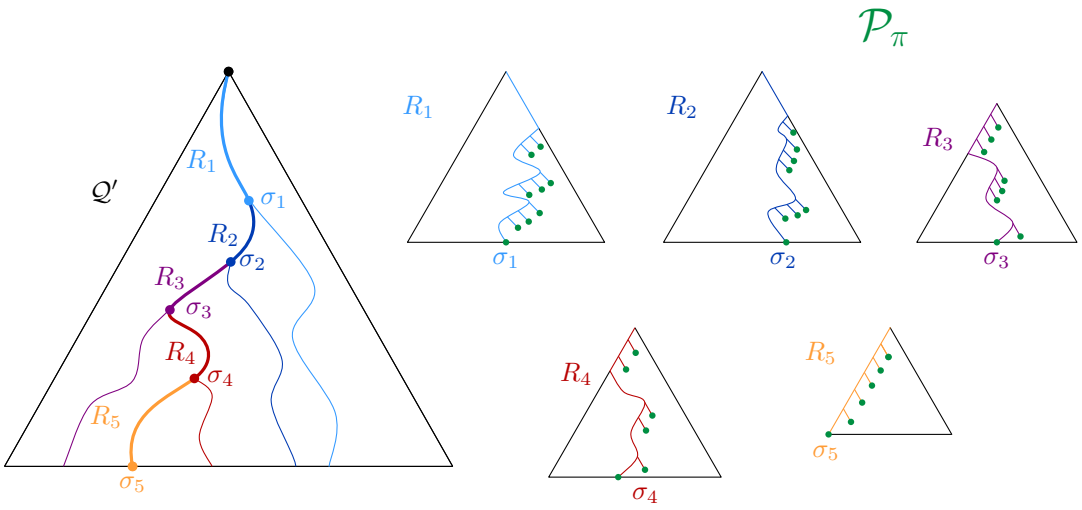

Figure 2 Illustration of Lemma 3.1. Left we see the decomposition of π into $O(\log n)$ heavy paths $R_1, \ldots R_k$. On the right the vertices defining $\mathcal{P}_\pi$ are depicted in green.

Given a constant $d \in \mathbb{N}$, let $\mathcal{C}_d$ be a set of d cones with common apex in the origin, each with opening angle $\frac{2\pi}{d}$, covering the plane. Let $P \in \mathcal{P}$ be a canonical path with smallest cell σ and largest cell τ. Then we denote the copy of the cones shifted to the center $a(\sigma)$ of σ by $\mathcal{C}_d(P)$. For constants d_1 and d_2 to be fixed later, now define $d_1 + d_2 + 1$ *regions* for P. These regions are partitioned into *outer regions*, *middle regions* and the *inner region*. The outer regions are obtained by considering $\mathcal{C}_{d_1}(P)$ and taking the intersection of each cone with the annulus with center $a(\sigma)$, outer radius $\frac{5}{2}|\sigma| + 2|\tau|$ and inner radius $\frac{5}{2}|\sigma|$. The middle regions are defined in a similar way, by taking the intersection of the cones in $\mathcal{C}_{d_2}(P)$ with the annulus with center $a(\sigma)$, outer radius $\frac{5}{2}|\sigma|$ and inner radius $|\sigma|$. Finally the inner region is defined as the disk with center $a(\sigma)$ and radius $|\sigma|$, see Figure 3 for an illustration. We call the set of the regions defined this way for all canonical paths $\mathcal{A}$.

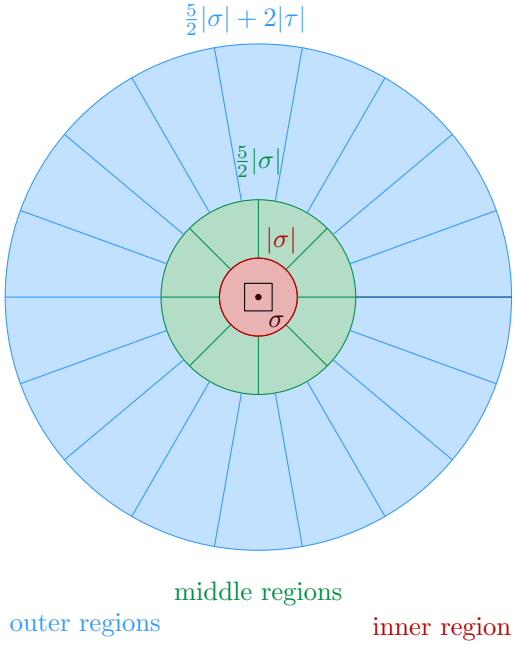

Figure 3 The regions defined by the smallest cell σ of a path

Based on these regions we define a sparse proxy graph H, which will be used to represent the connectivity in $D(S)$. The graph H has the vertex set $V = S \cup \mathcal{A}$. We will not strictly distinguish between the vertices of H and their associated sites and regions. With each region A we assign two sets $S_1(A) \subseteq S$ and $S_2(A) \subseteq S$. The set $S_1(A)$ will contain all sites contained in A whose are comparable to the size of the cells in the canonical path associated with A. The choice of the inner and outer radii in the definition of the regions together with our definition of comparable ensure that the disk graph induced by $S_1(A)$ is a clique. The set $S_2(A)$ contains sites with small radius which are contained in the smallest cell of the canonical path associated with A. These sites are chosen is such a way, that all edges in $D(S)$ between a site $s \in S_2(A)$ and $S_1(A)$ are represented with a single edge in H. As the sites in $S_1(A)$ form a clique this will not add a connection between sites in H which is not present in $D(S)$. Note that a site s will be contained in multiple sets $S_1(A)$ and $S_2(A)$ as it can lie in multiple regions and cells. Below we will give the details on the definition of $S_1(A)$ and $S_2(A)$. In Lemma 3.2 we show that the sites in $S_1(A)$ indeed form a clique. Then, in Lemma 3.3 we show that H accurately represents the connectivity of $D(S)$ and finally, as part of Lemma 4.1, we show that H is indeed sparse.

Now we give the details. The edge set of H is divided into two sets E_1 and E_2. The set E_1 is defined based on the sites in $S_1(A)$, whereas the set E_2 is defined based on $S_1(A)$. Let A be a region, where σ and τ are the smallest and largest cells respectively in the corresponding canonical path. If A is an outer region, let $S_1(A)$ be the set of all sites $t \in A$ with $|\sigma| \leq r_t \leq 2|\tau|$ and $\|a(\sigma)t\| \leq r_t + \frac{5}{2}|\sigma|$. On the other hand, if A is a middle or inner region, $S_1(A)$ consists of all sites $t \in A$ with $|\sigma| \leq r_t \leq 2|\tau|$. The edge set E_1 now consists of edges between A and all sites in $S_1(A)$.

Additionally we define a set $S_2(A)$ for the region. If A is an inner region, the set $S_2(A)$ consists of all sites s in σ with $r_s < |\sigma|$ which intersect at least one site in $S_1(A)$. In the other case, the set $S_2(A)$ contains all sites s in σ with $r_s \leq 2|\sigma|$, which are again intersecting at least one site in $S_1(A)$. The set E_2 now consists of edges between A and the sites in $S_2(A)$.

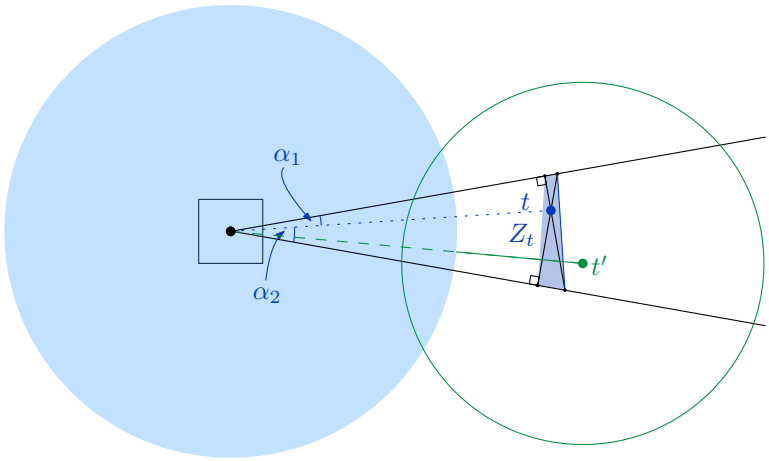

Figure 4 The part of the line segment $a(\sigma)t'$ in t' intersects the boundary of Z_t.

In order to show that this graph accurately represents the connectivity of the underlying disk graph $D(S)$ we first show that all sites within the same set $S_1(A)$ form a clique.

▶ **Lemma 3.2.** *Suppose that $d_1 \geq 18$ and $d_2 \geq 8$. Then, for any region $A \in \mathcal{A}$, the associated sites in $S_1(A)$ form a clique in $D(S)$.*

Proof (Sketch). As before, let σ be the smallest cell on the canonical path associates with A. If A is an inner or a middle regions this follows from considering the diameter of the regions. In both cases the diameter is at most $2|\sigma|$. As the sites in $S_1(A)$ have radius at least $|\sigma|$, the claim follows.

The case where A is an outer region is a bit more complicated. For each site $t \in S_1(A)$ we consider the two line segments going through t which are perpendicular to the lines defining the cone of A, and call the convex hull of these line segments Z_t, see Figure 4. Basic trigonometry shows that $Z_t \subseteq D_t$. Let $t' \in S_1(A)$ be a site with larger distance to $a(\sigma)$ than t. It can be argued that $t' \in S_1(A)$ either lies in Z_t or the part of the line segment $a(\sigma)t'$ which lies in D'_t intersects Z_t. In both cases it follows that the edge $\{t, t'\}$ exists in $D(S)$. ◀

Using Lemma 3.2 we can now show that H represents the connectivity in $D(S)$.

▶ **Lemma 3.3.** *Two sites $s, t \in S$ are connected in H if and only if they are connected in $D(S)$.*

Proof. First, we show that if s and t are connected in H they are also connected in $D(S)$. The graph H is bipartite, thus it suffices to show that if two sites u, u' are connected to the same region $A \in \mathcal{A}$, they are also connected in $D(S)$. This follows directly from Lemma 3.2: if u and u' are both in $S_1(A)$ they are part of the same clique and thus adjacent. If on the other hand $u \in S_2(A)$ or $u' \in S_2(A)$, the definition of $S_2(A)$ implies that $S_1(A)$, and the corresponding clique, is not empty. Thus there is a path from u through $S_1(A)$ to u'.

Now we consider two sites connected in $D(S)$ and show that they are also connected in H. If suffices to show that if s and t are connected by an edge in $D(S)$, they are connected to the same region $A \in \mathcal{A}$. Assume without loss of generality that $r_s \leq r_t$. Refer to Figure 5 for a depiction of the following argument. By the observation made above, there is a cell ρ in $N(t)$ which contains s. Consider the path π in $\mathcal{Q}'$ from the root to the smallest cell γ such that $s \in \gamma$ and $r_s \leq 2|\gamma|$. Then ρ lies on this path π. Let $\mathcal{P}_\pi$ be the decomposition of π

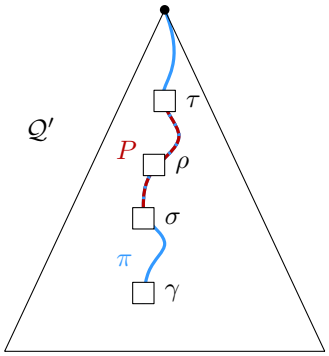

Figure 5 The cell γ is the smallest cell such that $r_s \leq 2|\gamma|$. The canonical path P contains ρ.

into canonical paths as defined in Lemma 3.1 and let P be the path containing ρ. Again let σ and τ be the smallest and largest cell on P respectively. By the definition of P we have $\gamma \subseteq \sigma \subseteq \rho \subseteq \tau$. As $\{s, t\}$ is an edge in $D(S)$, we have $\|st\| \leq r_s + r_t \leq 2|\sigma| + 2|\tau|$ and thus $\|a(\sigma)t\| \leq \frac{5}{2}|\sigma| + 2|\tau|$. This implies that t lies in a region A defined by P. If A is an inner region and $|\sigma| \leq r_s \leq 2|\sigma|$ then s is also part of $S_1(A)$ by definition, in the other case s is contained in $S_2(A)$. In both cases it follows that s and t are both connected to A. ◀

4 The Data Structure

The main idea of the data structure is to store H in a Holm et al. data structure $\mathcal{H}$ and use $\mathcal{H}$ to answer queries. The main challenge is to maintain H and $\mathcal{H}$ under deletion of sites, as well as efficiently preprocess the sites. The role of the components is shown in Figure 6.

▶ **Lemma 4.1.** *The graph H has $O(n)$ vertices, and $O(n \log^2 n)$ edges. The graph and a Holm et al. data structure $\mathcal{H}$ containing it can be constructed in $O(n \log^4 n)$ time. Within the same time bound we can construct the extended quadtree $\mathcal{Q}'$, the heavy path decomposition $\mathcal{R}$ and the binary search trees on the heavy paths.*

Proof (Sketch). The augmented compressed quadtree has $O(n)$ vertices. By adding appropriate *virtual sites* to the centers of suitable cells, it can be constructed in $O(n \log n)$ time. Constructing the heavy path decomposition on this tree takes time $O(n)$ by Lemma 2.1. The binary search trees defining the canonical paths also have a total $O(n)$ vertices and can be constructed in $O(n \log^2 n)$ overall time. As we define $O(1)$ regions for each canonical path, the bound on the number of vertices follows.

The number of edges follows from the total size of the sets $S_1(A)$ and $S_2(A)$. For the total size of the sets $S_1(A)$ consider a site t and its neighborhood $N(t)$. Each cell in $N(t)$ is contained in $O(\log n)$ canonical paths. In order to satisfy both the distance and the radius constraint for the set $S_1(A)$ of a region, its associated path has to contain a cell of $N(t)$ and thus t is contained in $O(\log n)$ sets $S_1(A)$.

A site s is contained in the sets $S_2(A)$ along the canonical paths which decompose the path π from the root to the smallest cell γ in $\mathcal{Q}'$ with $s \in \gamma$ and $r_s \leq 2|\gamma|$, again refer to Figure 5. By Lemma 3.1 there are $O(\log^2 n)$ such regions. Summing up over all sets $S_1(A)$ and $S_2(A)$ this results in $O(n \log^2 n)$ overall edges.

The sets $S_1(A)$ can be found by following the argument about the size and explicitly checking the conditions for each canonical path containing a given cell of $N(t)$. In order to find the sets $S_2(A)$ we first have to construct an additively weighted voronoi diagram on each

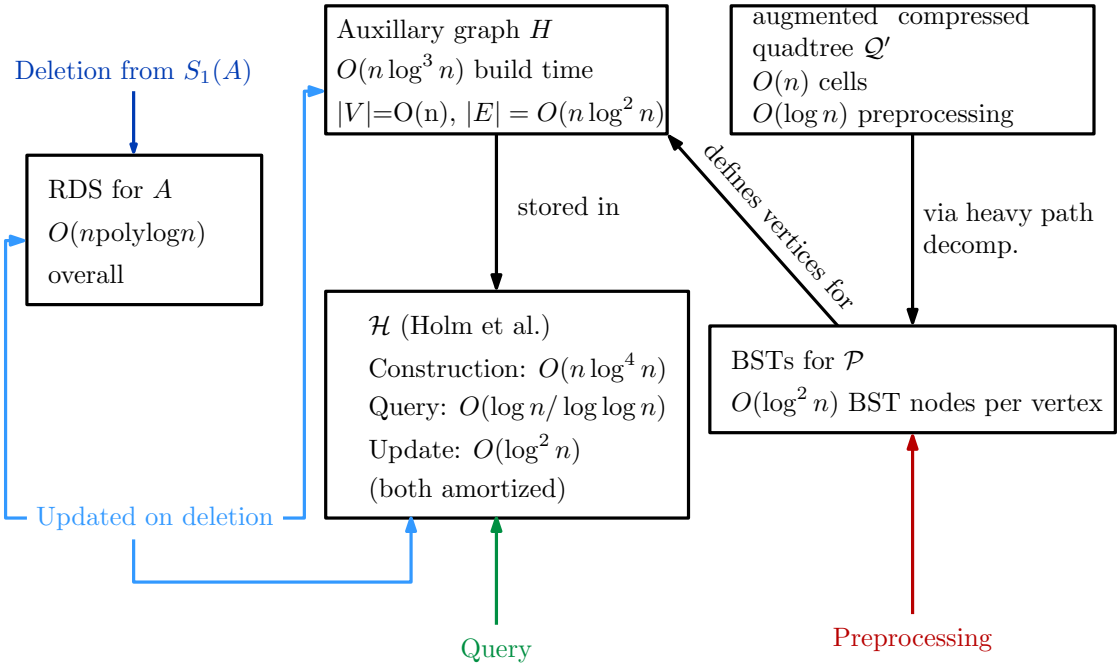

Figure 6 Components of the data structure and their connections

set $S_1(A)$. Assigning each site $t \in S_1(A)$ the weight $-r_t$, allows us to determine in $O(\log n)$ time if a site s intersects some site in $S_1(A)$. Performing this query for all regions along the path defined in the size argument yields all sets $S_2(A)$. Combining the steps for finding $S_1(A)$ and $S_2(A)$ finds all edges in $O(n \log^3 n)$ time. Inserting the edges one by one to the Holm et al. data structure $\mathcal{H}$ dominates the time, leading to $O(n \log^4 n)$ overall time. ◀

We simply perform connectivity queries on $\mathcal{H}$. To handle deletions, we build one reveal data structure (RDS) for each region $A \in \mathcal{A}$. We set $R' = S_1(A)$ and $B' = S_2(A)$ for each RDS. Let n' be the total number of sites stored in one of the data structures, then preprocessing, deleting some sites from B', deleting m' sites from R' and retrieving the revealed sites takes $O((n' \log^5 n + m' \log^9 n)\lambda_6(\log n))$ total time by Lemma 2.2.

When deleting a site s, we can safely delete it from all sets $S_2(A)$ containing it, together with the associated edges in E_2 and the sites stored in the RDS of A. When deleting a site from a set $S_1(A)$ however it is possible that some other sites have to be removed from the associated set $S_2(A)$. These sites are reported by the RDS on deleting s from R', allowing us to safely delete them from $S_2(A)$. Summing up we get the following result.

▶ **Theorem 4.2.** *The data structure described above answers connectivity queries in amortized time* $O\left(\frac{\log n}{\log \log n}\right)$ *with* $O((n \log^7 n + m \log^{11} n)\lambda_6(\log n))$ *overall expected update time for* m *deletions.*

References

1 Kevin Buchin, Maarten Löffler, Pat Morin, and Wolfgang Mulzer. Preprocessing imprecise points for Delaunay triangulation: Simplified and extended. *Algorithmica*, 61(3):674–693, 2011. doi:10.1007/s00453-010-9430-0.

2 Mark de Berg, Otfried Cheong, Marc van Kreveld, and Mark Overmars. *Computational Geometry: Algorithms and applications*. Springer-Verlag, Berlin, 3rd edition, 2008. doi:10.1007/978-3-540-77974-2.

3 Sariel Har-Peled. Quadtrees–hierarchical grids. In *Geometric Approximation Algorithms*, volume 173 of *Mathematical Surveys and Monographs*, chapter 2. American Mathematical Society, 2011. doi:10.1090/surv/173.

4 Jacob Holm, Kristian de Lichtenberg, and Mikkel Thorup. Poly-logarithmic deterministic fully-dynamic algorithms for connectivity, minimum spanning tree, 2-edge, and biconnectivity. *J. ACM*, 48(4):723–760, 2001. doi:10.1145/502090.502095.

5 Haim Kaplan, Alexander Kauer, Wolfgang Mulzer, and Liam Roditty. Sampling hyperplanes and revealing disks. Abstract submitted to EuroCG21., 2021.

6 Haim Kaplan, Wolfgang Mulzer, Liam Roditty, and Paul Seiferth. Dynamic connectivity for unit disk graphs. In *Proc. 32nd European Workshop on Computational Geometry (EWCG)*, 2016.

7 Daniel D. Sleator and Robert Endre Tarjan. A data structure for dynamic trees. *J. Comput. Syst. Sci.*, 26(3):362–391, 1983. doi:10.1016/0022-0000(83)90006-5.

8 Dan E. Willard and George S. Lueker. Adding range restriction capability to dynamic data structures. *J. ACM*, 32(3):597–617, 1985. doi:10.1145/3828.3839.