Approximate Shortest Paths and Distance Oracles in Weighted Unit-Disk Graphs

Timothy M. Chan
Department of Computer Science, University of Illinois at Urbana-Champaign, USA
tmc@illinois.edu

Dimitrios Skrepetos
Cheriton School of Computer Science, University of Waterloo, Canada
dskrepet@uwaterloo.ca

Abstract

We present the first near-linear-time $(1+\varepsilon)$-approximation algorithm for the *diameter* of a weighted unit-disk graph of n vertices, running in $O\left(n \log^2 n\right)$ time, for any constant $\varepsilon > 0$, improving the near-$O\left(n^{3/2}\right)$-time algorithm of Gao and Zhang [STOC 2003]. Using similar ideas, we can construct a $(1+\varepsilon)$-approximate *distance oracle* for weighted unit-disk graphs with $O(1)$ query time, with a similar improvement in the preprocessing time, from near $O\left(n^{3/2}\right)$ to $O\left(n \log^3 n\right)$. We also obtain new results for a number of other related problems in the weighted unit-disk graph metric, such as the radius and bichromatic closest pair.

As a further application, we use our new distance oracle, along with additional ideas, to solve the $(1+\varepsilon)$-approximate *all-pairs bounded-leg shortest paths* problem for a set of n planar points, with near $O\left(n^{2.579}\right)$ preprocessing time, $O\left(n^2 \log n\right)$ space, and $O\left(\log \log n\right)$ query time, improving thus the near-cubic preprocessing bound by Roditty and Segal [SODA 2007].

2012 ACM Subject Classification Theory of computation → Computational geometry, Theory of computation → Shortest paths

Keywords and phrases shortest paths, distance oracles, unit-disk graphs, planar graphs

Digital Object Identifier 10.4230/LIPIcs.SoCG.2018.24

1 Introduction

In this paper, we study shortest-path problems in *weighted unit-disk graphs*, i.e., intersection graphs of unit disks. More concretely, in such a graph, vertices correspond to a set S of planar points (specifically, the centers of the disks), and there is an edge between every two points of S at Euclidean distance at most one (of weight equal to that distance). These graphs have been widely used in many applications, such as modelling ad-hoc communication networks.

We are interested in various basic problems about shortest paths in such a weighted unit-disk graph G, notably:

- designing algorithms for computing a $(1+\varepsilon)$-approximation of various parameters of G, such as the *diameter* (i.e., $\max_{s,t \in S} d_G[s,t]$), the *radius* (i.e., $\min_{s \in S} \max_{t \in S} d_G[s,t]$), the *bichromatic closest pair distance* of two subsets $A, B \subset S$ (i.e., $\min_{a \in A,\, b \in B} d_G[a,b]$), et cetera; and,
- designing *approximate distance oracles*, i.e., data structures that support the following query: given any $s, t \in S$, quickly compute a $(1+\varepsilon)$-approximation of the s-to-t shortest-path distance in G, $d_G[s,t]$.

Besides practical motivation from wireless networks, this collection of problems is interesting from the theoretical perspective as well since it connects computational geometry with graph data structures – indeed, our new algorithms will draw on ideas from both areas.

Planar graph techniques. There has already been an extensive body of work devoted to distance oracles and shortest-path-related problems both for general and for planar graphs, the latter of which are of particular relevance to us. For example, Thorup [25] gave $(1+\varepsilon)$-approximate distance oracles for weighted, undirected planar graphs with $O(n \operatorname{polylog} n)$ preprocessing time and space, and $O(1)$ query time, for any constant $\varepsilon > 0$, while subsequent work [19, 18, 16, 10] gave improvements and examined the dependency of the hidden factors on ε. Weimann and Yuster [27] presented a $(1+\varepsilon)$-approximation algorithm for the diameter for weighted, undirected planar graphs, running in $O\left(n \log^4 n\right)$ time, for any constant $\varepsilon > 0$, improved later to $O\left(n \log^2 n\right)$ by Chan and Skrepetos [10], who also reduced the ε-dependency from exponential to polynomial (there has also been exciting recent breakthrough on *exact* algorithms for diameter and distance oracles in planar graphs, by Cabello [5] and subsequent researchers [15, 11, 14]).

All the above approximation results for planar graphs rely heavily on the concept of *shortest-path separator*: a set of shortest paths with common root, such that the removal of their vertices decomposes the graph into at least two disjoint subgraphs. Unfortunately, such separators do not seem directly applicable to unit-disk graphs, and not only because the latter may be dense. Indeed, by grid rounding, we can construct a sparse weighted graph $\widehat{G}$, such that it (i) approximately preserves distances in the original unit-disk graph G (e.g., see the proof of Lemma 2), and (ii) is "nearly planar", in the sense that each edge intersects at most a constant number of other edges. However, even for such a graph, it is not clear how to define a shortest-path separator that divides it cleanly into an inside and an outside because edges may "cross" over the separator. At least one prior paper [28] worked on extending shortest-path separators to unit-disk graphs, but the construction was complicated and achieved only constant approximation factors.

Gao and Zhang's WSPD technique. In a seminal paper, Gao and Zhang [13] obtained the first nontrivial set of results on shortest-path problems in weighted unit-disk graphs, by adapting a familiar technique in computational geometry – namely, the *well-separated pair decomposition* (WSPD), introduced by Callahan and Kosaraju [7] for addressing proximity problems in the Euclidean (or L_p) metric and has since found countless applications. Gao and Zhang proposed a new variant of WSPDs for the weighted unit-disk graph metric and showed that any n-point set in two dimensions has a WSPD of near-linear ($O(n \log n)$) size under the new definition. Consequently, they obtained a $(1+\varepsilon)$-approximate distance oracle with $O(n \log n)$ size and $O(1)$ query time, for any constant $\varepsilon > 0$. Unfortunately, its preprocessing time, $O\left(n^{3/2}\sqrt{\log n}\right)$, is quite high and becomes the bottleneck when the technique is applied to offline problems such as computing the diameter.

However, the issue is not constructing the WSPD itself, which can be done in near-linear time, but computing the shortest-path distances of a near-linear number of vertex pairs in the "nearly planar" graph $\widehat{G}$ mentioned above, which takes almost $n^{3/2}$ time, by adapting a known exact distance oracle for planar graphs [1] (noting that $\widehat{G}$ has balanced separators [22, 12]). Cabello [4] has given an improved algorithm for computing multiple distances in planar graphs, and if it could be adapted here, the running time would be reduced to around $n^{4/3}$. However, near-linear time still seems out of reach with current techniques.

Gao and Zhang [13] observed that the preprocessing time can be made near-linear when the approximation factor is a certain constant (about 2.42), but this improvement does not apply to $1+\varepsilon$ approximation factor and has no new implication to the diameter problem (for which a near-2-approximation is easy by running a single-source shortest paths algorithm).

New results. In Section 2, we give the first near-linear-time algorithm to compute a $(1+\varepsilon)$-approximation of the diameter of a weighted unit-disk graph, running in $O\left(n \log^2 n\right)$ time, for any constant $\varepsilon > 0$ (the dependencies of the hidden factors on ε are polynomial). A similar result holds for $(1+\varepsilon)$-approximate distance oracles: we obtain $O\left(n \log^3 n\right)$ preprocessing time, $O\left(n \log n\right)$ space, and $O(1)$ query time. We thus answer one of the main questions left open in Gao and Zhang's paper, while also apply our techniques to related problems.

Our approach is conceptually simple: we just go back to known shortest-path separator techniques for planar graphs [25, 18]!

But how do we get around the issue that unit-disk graphs do not have nice path separators? We first find a *spanner* subgraph H that is planar and has constant approximation/stretch factor (fortunately, such spanners are known to exist in unit-disk graphs [21] and they were also used by Gao and Zhang [13]) and then then apply divide-and-conquer over the shortest-path separator decomposition tree for H instead of G.

Although the above plan may sound obvious in hindsight, the details are tricky to get right. For example, how could the use of a spanner with $O(1)$ approximation factor eventually lead to $1+\varepsilon$ approximation factor? The known divide-and-conquer approaches for planar graphs select a small number of vertices, called *portals*, along each separator and compute distances from each with a Single-Source Shortest Paths algorithm; that works well because a shortest path in a planar graph crosses a separator only at vertices. In our case, however, we need to use the original (non-planar, unit-disk) graph G when computing distances from portals, but therein a shortest path could "cross" the separator over an edge. We show that we can nevertheless re-route such a path to pass through a separator vertex without increasing the length by much, by using the fact that H is a $O\left(1\right)$-spanner.

Application to all-pairs bounded-leg shortest paths. In the last part of the paper, as a further application, we employ our new distance oracle, along with additional ideas, to solve the $(1+\varepsilon)$-approximate *All-Pairs Bounded-Leg Shortest Paths* (apBLSP) problem. Given a set S of n planar points, we define $G_{\leq L}$ to be the subgraph of the complete Euclidean graph of S that contains only edges of weight at most L. Then, we want to preprocess S, such that given two points $s, t \in S$ and any positive number L, we can quickly compute a $(1+\varepsilon)$-approximation of the s-to-t shortest path in $G_{\leq L}$ (i.e., the shortest path under the restriction that each leg of the trip has length bounded by L) or its length. To see the connection of apBLSP with the earlier problems, note that, for each fixed L, $G_{\leq L}$ is a weighted unit-disk graph, after rescaling the radii. One important difference, however, is that L is not fixed in apBLSP, and we want to answer queries for any of the $\binom{n}{2}$ combinatorially different L's.

Bose et al. [2] introduced that problem in 2003 and gave a data structure for it with $O\left(n^5\right)$ preprocessing time, $O\left(n^2 \log n\right)$ space, and $O\left(\log n\right)$ query time, for any constant $\varepsilon > 0$, while Roditty and Segal [24] improved the preprocessing time to roughly $O\left(n^3\right)$ and the query time to $O\left(\log \log n\right)$.

In Section 3, we apply our $(1+\varepsilon)$-approximate distance oracle for weighted unit-disk graphs, along with additional new ideas, to obtain the first method to break the cubic preprocessing barrier: we can obtain roughly $O\left(n^{8/3}\right)$ preprocessing time, while keeping

$O\left(n^2 \log n\right)$ space and $O\left(\log \log n\right)$ query time. With fast matrix multiplication, we can further reduce the preprocessing time to $O\left(n^{2.579}\right)$, assuming a polynomial bound on the *spread*, i.e., the ratio of the maximum to the minimum Euclidean distance over all pairs of points in V.

2 Approximate diameter and distance oracles

Let S be a set of planar points whose weighted unit-disk graph G has diameter Δ. A key subproblem in both (i) computing a $(1+\varepsilon)$-approximation of the diameter of G and (ii) building a $(1+\varepsilon)$-approximate distance oracle for it is the construction of a distance oracle with *additive stretch* $O\left(\varepsilon\Delta\right)$: a data structure, such that, given any $s, t \in S$, we can quickly compute a value $\widetilde{d}$ with $d_G[s,t] \leq \widetilde{d} \leq d_G[s,t] + O\left(\varepsilon\Delta\right)$. We describe our solution for that subproblem in Section 2.2, after giving two preliminary ingredients in Section 2.1, and then show, in Section 2.3, how to employ it, along with existing techniques, to address the two original problems.

2.1 Preliminaries

The first ingredient we need is the existence of a planar spanner with constant stretch factor in any weighted unit-disk graph.

▶ **Lemma 1** (Planar spanner). *Given a set S of n planar points, we can find, in $O\left(n \log n\right)$ time, a planar spanning subgraph H of its weighted unit-disk graph G, such that, for every $s, t \in S$, $d_G[s,t] \leq d_H[s,t] \leq c d_G[s,t]$, where c is some constant.*

Li, Calinescu, and Wan [21] proved the above lemma with $c = 2.42$ by simply building the Delaunay triangulation of the given points and discarding edges of weight more than one; however, the analysis of the stretch factor c is nontrivial.

The second ingredient is an efficient algorithm for the Single-Source Shortest Paths (SSSP) problem in weighted unit-disk graphs, where the currently best exact result, due to Cabello and Jejčič [6], requires $O\left(n \log^{12+o(1)} n\right)$ time and employs complicated dynamic data structures for additively-weighted Voronoi diagrams [8, 17]. For our purposes though, it suffices to consider the $(1+O\left(\varepsilon\right))$-approximate version of the problem instead, i.e., given a set of points S and a source $s \in S$, compute, for each $t \in S$, a path of length $\widetilde{d}[s,t]$, such that $d_G[s,t] \leq \widetilde{d}[s,t] \leq (1+O\left(\varepsilon\right)) d_G[s,t]$, where G is the weighted unit-disk graph of S. Our algorithm first finds a sparse graph $\widehat{G}$ that $(1+O\left(\varepsilon\right))$-approximately preserves distances in G (i.e., for any $s, t \in S$, there are vertices p_s, p_t of $\widehat{G}$, such that $\delta_G[s,t] \leq \delta_{\widehat{G}}[c_s, c_t] \leq (1+O\left(\varepsilon\right)) d_G[s,t]$) and then runs Dijkstra's algorithm therein; sparsification in weighted unit-disk graphs has been used before (e.g., see [13, Section 4.2]).

▶ **Lemma 2** (Approximate SSSP). *Given a set S of n planar points, we can solve the $(1+O\left(\varepsilon\right))$-approximate SSSP problem in its weighted unit-disk graph G in $O\left((1/\varepsilon)^2 n \log n\right)$ time.*

Proof. First, we build a uniform grid of side length ε and construct a sparse weighted graph $\widehat{G}$ by placing a vertex at each non-empty grid cell and an edge between every two such cells c and c' iff there exist points $p \in c$ and $p' \in c'$ with $\|pp'\| \leq 1$; the weight of that edge is equal to the maximum Euclidean distance of c and c'. Each grid cell has at most $O\left((1/\varepsilon)^2\right)$ neighbors, so $\widehat{G}$ has at most $O\left((1/\varepsilon)^2 n\right)$ edges and can be constructed in $O\left((1/\varepsilon)^2 n \log n\right)$ time, by using a Euclidean bichromatic closest pair algorithm [23] over $O\left((1/\varepsilon)^2 n\right)$ pairs of grid cells.

Let s and t be two points of S; if $\|st\| \leq 1$, we can trivially return $\|st\|$. Else, let $p_0 p_1 \cdots p_\ell$, with $p_0 = s$ and $p_\ell = t$, be the shortest path in G from s to t. Two consecutive edges therein have lengths whose sum is at least one because otherwise we could take a short-cut and obtain a shorter path; thus, $d_G[s,t] \geq \lfloor \ell/2 \rfloor$. We construct a path $c_0 c_1 \cdots c_\ell$ in $\widehat{G}$, where each c_i is the cell that contains p_i. Since, for each c_i, c_{i+1}, $\|p_i p_{i+1}\| \leq d_{\widehat{G}}[c_i, c_{i+1}] \leq \|p_i p_{i+1}\| + O(\varepsilon)$, it follows that $d_G[s,t] \leq d_{\widehat{G}}[c_0, c_\ell] \leq d_G[s,t] + O(\varepsilon\ell) \leq (1 + O(\varepsilon)) d_G[s,t]$.

Thus, given a source $s \in S$, we can invoke Dijkstra's algorithm in $\widehat{G}$ to compute, for each $t \in S$, a value $d_{\widehat{G}}[c_s, c_t]$, such that $\delta_G[s,t] \leq \delta_{\widehat{G}}[c_s, c_t] \leq (1 + O(\varepsilon)) d_G[s,t]$, where c_s and c_t are the grid cells that contain s and t, respectively. We can easily modify our algorithm to also find, for each $t \in S$, an s-to-t path in G of length $\delta_{\widehat{G}}[c_s, c_t]$, by appending s and t at the ends of the s-to-t shortest path in $\widehat{G}$ and replacing each c_i and c_{i+1} with the bichromatic closest pair of $((S \cap c_i), (S \cap c_{i+1}))$ in G (which has been found while constructing $\widehat{G}$). ◀

2.2 Distance oracles with $O(\varepsilon\Delta)$ additive stretch

We describe now a distance oracle with additive-stretch for an arbitrary weighted graph $G = (V, E)$ of n vertices and of diameter Δ that has the following properties, which are the only ones needed from weighted unit-disk graphs.

- **(I)** There exists a planar *c-spanner* H of G, for some constant c.
- **(II)** For any induced subgraph of G with n' vertices, the $(1+\varepsilon)$-approximate SSSP problem can be solved in $T(n')$ time, for some function $T(\cdot)$, such that $T(n')/n'$ is nondecreasing.
- **(III)** Every edge weight in G is at most $\varepsilon\Delta$.

If G is a weighted unit-disk graph, Lemmas 1 and 2 imply (I) and (II), respectively, where $c = 2.42$ and $T(n') = O\left((1/\varepsilon)^2 n' \log n'\right)$, and (III) holds as long as $\Delta \geq 1/\varepsilon$.

Shortest-path separators in H. Although G may not necessarily have nice shortest-path separators, we know that H does, by planarity. Thus, we apply a known shortest-path separator decomposition therein, namely the version of Kawarabayashi, Sommer, and Thorup [18, Section 3.1], paraphrased for our purposes. Specifically, we can compute in $O(n \log n)$ time a *decomposition tree* $\mathcal{T}$ with the following properties.

- $\mathcal{T}$ has $O(1)$ degree and $O(\log n)$ height.
- Each node μ of $\mathcal{T}$ is associated with a subset $V^{(\mu)} \subseteq V$. The subsets $V^{(\nu)}$ over all children ν of μ are disjoint and contained in $V^{(\mu)}$. If μ is the root, $V^{(\mu)} = V$; if μ is a leaf, $V^{(\mu)}$ has $O(1)$ size.
- Each non-leaf node μ of $\mathcal{T}$ is associated with a set of $O(1)$ paths, called *separator paths*, which are classified as "internal" and "external". The internal separator paths cover precisely all vertices of $V^{(\mu)} - \bigcup_{\text{child } \nu \text{ of } \mu} V^{(\nu)}$, while the external are outside of $V^{(\mu)}$.
- For each child ν of a non-leaf node μ, every neighbor of the vertices of $V^{(\nu)}$ in H is either in $V^{(\nu)}$ or in one of the (internal or external) separator paths at μ.
- Each separator path is a shortest path in H and, in particular, has length at most the diameter $\Delta(H)$ of H (which is at most $c\Delta$).

Our data structure. To construct an additive oracle with $O(\varepsilon\Delta)$ stretch for G, we construct the above decomposition tree $\mathcal{T}$ and augment it with extra information, as follows. Let μ be an internal node of $\mathcal{T}$ and σ one of its internal separator paths; since σ has length at

most $\Delta(H) \leq c\Delta$, we can select, with a linear walk, a set of $O(1/\varepsilon)$ vertices thereon, called *portals*, such that each consecutive pair of them is at distance at most $\varepsilon\Delta$ on it.

Let $P^{(\mu)}$ denote the set of all portals over all internal separator paths at a non-leaf node μ of $\mathcal{T}$. For each such node and for each $p \in P^{(\mu)}$ and $v \in V^{(\mu)}$, we invoke $O(1/\varepsilon)$ times the SSSP algorithm from Property (II) to compute a $(1+\varepsilon)$-approximation, $\widetilde{d}_\mu[p,v]$, of the shortest path distance from p to v in the subgraph of G induced by $V^{(\mu)}$. Then, for each leaf μ, we just find and store all pairwise distances in the subgraph of G that is induced by $V^{(\mu)}$. Overall, our oracle requires $O((1/\varepsilon)T(n) \cdot \log n)$ preprocessing time and $O((1/\varepsilon)n \cdot \log n)$ space.

Query algorithm. Given two vertices $s, t \in V$, we first identify all $O(\log n)$ nodes μ in $\mathcal{T}$, such that both $s \in V^{(\mu)}$ and $t \in V^{(\mu)}$ (by trivially starting from the root and going down the tree along a path). For each such non-leaf node μ, we compute, in $O(1/\varepsilon)$ time, a value $\widetilde{\delta}_\mu[s,t] = \min_{p \in P^{(\mu)}} \{\widetilde{d}_\mu[s,p] + \widetilde{d}_\mu[p,t]\}$. If μ is a leaf, $\widetilde{d}_\mu[s,t]$ is the exact shortest path distance in the subgraph of G induced by $V^{(\mu)}$ (which we have already computed). Finally we return the minimum, $\widetilde{\delta}[s,t]$, over all $\widetilde{\delta}_\mu[s,t]$. The total query time is $O((1/\varepsilon)\log n)$.

Stretch analysis. We want to prove that for any $s, t \in V$, the value, $\widetilde{d}[s,t]$, that our oracle returns is such that $d_G[s,t] \leq \widetilde{d}[s,t] \leq d_G[s,t] + O(\varepsilon\Delta)$. The left side of the inequality clearly holds because $\widetilde{d}[s,t]$ corresponds to the length of an s-to-t path in a subgraph of G. To prove the right side, let π be the shortest s-to-t path in G, and let μ be the lowest node in $\mathcal{T}$, such that all vertices of π lie in $V^{(\mu)}$; we assume that μ is a non-leaf node (otherwise we have already computed $d_G[s,t]$ exactly).

Although π is a path in the (not necessarily planar) graph G, not H, we show that it is possible to re-route it to pass through a vertex on a separator path of μ without increasing its length by much.

▶ **Claim 3** (Detour through a separator vertex)**.** *There exists an s-to-t path π' in G that (i) passes through some vertex w on a separator path of μ, (ii) uses only vertices of $V^{(\mu)}$ (except maybe for w itself) and (iii) has length at most $d_G[s,t] + 2c\varepsilon\Delta$.*

Proof. We assume that none of the vertices on π lie on a separator path of μ because otherwise we can just set $\pi' = \pi$. Let ν be the child of μ with $s \in V^{(\nu)}$, let u be the last vertex on π that lies in $V^{(\nu)}$ (note that $u \neq t$, by definition of μ), and let v be the next vertex after u thereon. By (I), there is a path $\pi_{u,v}$ from u to v in H of length at most $c \cdot$ (the weight of uv), which is at most $c\varepsilon\Delta$ by (III). Let w be the first vertex on $\pi_{u,v}$ that lies outside of $V^{(\nu)}$ (which exists since v is outside of $V^{(\nu)}$); then, from the fourth property of $\mathcal{T}$, we know that w must be on an (internal or external) separator path σ of μ. Thus, we set π' to be the path that goes from s to u along π, then from u to w along $\pi_{u,v}$ (which uses only vertices in $V^{(\nu)}$ as intermediates), then back from w to u along $\pi_{u,v}$, and finally from u to t along π. See Figure 1(a) (where σ is internal) and 1(b) (where σ is external). ◀

Next, we note how to further re-route π to pass through a portal.

▶ **Claim 4** (Detour through a portal)**.** *There exists another s-to-t path π'' in G that (i) passes through a portal p on a separator path σ' of μ', where μ' is some ancestor of μ, (ii) uses only vertices of $V^{(\mu')}$, and (iii) has length at most $d_G[s,t] + (2c+2)\varepsilon\Delta$.*

Proof. Let w be as in Claim 3, and let μ' be the lowest ancestor of μ, such that $w \in V^{(\mu')}$ (notice that if $w \in V^{(\mu)}$, $\sigma = \sigma'$). Then w must be on an internal separator path σ' in μ'

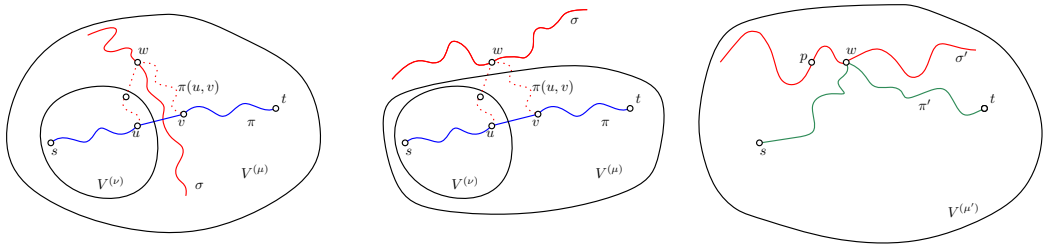

Figure 1 Detour through a vertex of a separator path σ in Claim 3, where σ may be internal, as in (a), or external, as in (b); detour through a portal in Claim 4 in (c).

(whose existence is guaranteed by the third property of $\mathcal{T}$). Let p be the portal on σ' that is closest to w, so the p-to-w distance on σ' is at most $O(\varepsilon\Delta)$. We set π'' to be the path the goes from s to w along π', then from w to p along σ' and back from p to w, and, finally, from w to u along π'. See Figure 1(c). ◀

Let μ' be as in Claim 4. It follows that $\widetilde{\delta}[s,t] \leq \widetilde{\delta}_{\mu'}[s,t] \leq \widetilde{d}_{\mu'}[s,p] + \widetilde{d}_{\mu'}[p,t] \leq d_G[s,t] + O(\varepsilon\Delta)$.

▶ **Theorem 5** (General Additive-Stretch Distance Oracle). *Given a weighted graph of n vertices and of diameter Δ that satisfies Properties (I)–(III), we can construct for it, in $O((1/\varepsilon)T(n)\log n)$ time, a distance oracle of $O(\varepsilon\Delta)$ additive stretch, $O((1/\varepsilon)n\log n)$ space, and $O((1/\varepsilon)\log n)$ query time.*

As we saw earlier, weighted unit-disk graphs satisfy Properties (I)-(III), thus we have the following theorem.

▶ **Corollary 6** (Additive-Stretch Distance Oracle in Unit-Disk Graphs). *Given a set S of n planar points, such that the weighted unit-disk graph of S has diameter $\Delta \geq 1/\varepsilon$, we can construct for the latter, a distance oracle of $O(\varepsilon\Delta)$ additive stretch, $O\left((1/\varepsilon)^3 n\log^2 n\right)$ preprocessing time, $O((1/\varepsilon)n\log n)$ space, and $O((1/\varepsilon)\log n)$ query time.*

2.3 Applications

We now describe how to employ Corollary 6 to compute a $(1+\varepsilon)$-approximation of the diameter of a unit-disk graph and how to build a $(1+\varepsilon)$-approximate distance oracle for it.

Approximate diameter. To approximate the diameter of a weighted unit-disk graph, we use the following lemma, implied by Gao and Zhang's WSPD-based technique [13, Corollary 5.2].

▶ **Lemma 7** (Via Well-Separated Pair Decomposition). *Given a set S of n planar points, we can find a set of $O\left((1/\varepsilon)^4 n\log n\right)$ pairs of them in $O\left((1/\varepsilon)^4 n\log n\right)$ time, such that the shortest-path distance between any two vertices in the weighted unit-disk graph of S can be $(1+\varepsilon)$-approximated by the shortest-path distance between one of these pairs, which can be found in $O(1)$ time.*

First we compute in $O(n\log n)$ time [23] the Euclidean diameter, Δ_0, of S; if $\Delta_0 \geq 1/\varepsilon$, then $\Delta \geq 1/\varepsilon$, and, to compute a $(1+\varepsilon)$-approximation of Δ, we can query the oracle of Corollary 6 of $O(\varepsilon\Delta)$ additive stretch with all $O\left((1/\varepsilon)^4 n\log n\right)$ pairs of Lemma 7 and return

the maximum; thus the approximation factor is $1 + O(\varepsilon)$. The total time required for this case is $O\left(((1/\varepsilon)^4 n \log n \cdot (1/\varepsilon) \log n\right)$.

If $1 < \Delta_0 < 1/\varepsilon$, the problem is more straightforward because we can construct the sparsified graph $\widehat{G}$ from the proof of Lemma 2, which preserves distances approximately, and then run a standard All-Pairs Shortest Paths (APSP) algorithm therein. Since $\widehat{G}$ has $\widehat{n} = O\left((\Delta_0/\varepsilon)^2\right) = O\left((1/\varepsilon)^4\right)$ vertices and $\widehat{m} = O\left((1/\varepsilon)^2 \widehat{n}\right) = O\left((1/\varepsilon)^6\right)$ edges, we need $O\left(\widehat{n}^2 \log \widehat{n} + \widehat{m}\widehat{n}\right) = O\left((1/\varepsilon)^{10}\right)$ time for this case. Finally, if $\Delta_0 < 1$, the unit-disk graph is a complete Euclidean graph, so we just return Δ_0.

▶ **Theorem 8** (Approximate Diameter). *Given a set S of n planar points, we can compute, in $O\left((1/\varepsilon)^5 n \log^2 n + (1/\varepsilon)^{10}\right)$ time, a $(1+\varepsilon)$-approximation of the diameter of the weighted unit-disk graph of S.*

▶ Remark. Employing a WSPD is not essential here, as we could combine our techniques with those of Weimann and Yuster for planar graphs [27], increasing, though, the ε-dependency to $2^{O(1/\varepsilon)}$.

Approximate distance oracles. To build a distance oracle of $(1+\varepsilon)$-approximation factor for a weighted unit-disk graph, we employ the oracle of Corollary 6 of $O(\varepsilon\Delta)$ additive stretch as a building block in a known technique called *sparse neighborhood covers*. We use sparse neighborhood covers result of Busch et al. [3] for planar graphs, whose construction time bound was given by Kawarabayashi et al. [18].

▶ **Lemma 9** (Sparse neighborhood cover). *Given a weighted planar graph H of n vertices and a value r, we can construct, in $O(n \log n)$ time, a collection of subsets V_i of V, such that (i) the diameter of the subgraph of H induced by each V is $O(r)$, (ii) every vertex resides in $O(1)$ subsets, and (iii) for every vertex v, the set of all vertices at distance at most r from v in H is contained in at least one of the V_i's.*

Let G be the weighted unit-disk graph of a set S of n planar points, and let H be an $O(1)$-planar spanner of G. Every shortest path distance in G is upper bounded by n, so we first apply the above lemma to H for each value of $r \in \{2^0, 2^1, \ldots, 2^{\log n}\}$, thus obtaining collections of subsets $V_i^{(r)}$, and then build the distance oracle of Corollary 6 for the weighted unit-disk graph of each $V_i^{(r)}$. The total preprocessing time and space over all $O(\log n)$ choices of r is $O\left(\log n \cdot (1/\varepsilon)^3 n \log^2 n\right)$ and $O(\log n \cdot (1/\varepsilon) n \log n)$, respectively. Given $s, t \in S$, we consider each r and each subset $V_i^{(r)}$ that contains both s and t, query the oracle for $V_i^{(r)}$, and return the minimum. The total query time over all $O(\log n)$ choices of r and $O(1)$ choices of $V_i^{(r)}$ (Lemma 9(ii)) is $O(\log n \cdot (1/\varepsilon) \log n)$.

If $d_G[s,t] \geq 1/\varepsilon$, let $r \geq c/\varepsilon$ be such that $d_G[s,t] \in (r/2c, r/c]$. Then, each vertex on the shortest path from s to t in G is at distance at most $c d_G[s,t] \leq r$ from s in H, so it is contained in a common subset $V_i(r)$, and we approximate $d_G[s,t]$ with an additive error of $O(\varepsilon r) = O(\varepsilon d_G[s,t])$, obtaining thus $1 + O(\varepsilon)$ approximation factor.

If $1 < d_G[s,t] < 1/\varepsilon$, we simply build the sparsified graph $\widehat{G}$ from the proof of Lemma 2, which preserves distances approximately, and, from every vertex, we pre-compute the distances to all grid cells at Euclidean distance at most $1/\varepsilon$, by running Dijkstra's algorithm on a subgraph of $\widehat{G}$ with $n' = O\left(1/\varepsilon\right)^4)$ vertices and $O\left((1/\varepsilon)^2 n'\right) = O\left((1/\varepsilon)^6\right)$ edges, in $O\left((1/\varepsilon)^6 \log(1/\varepsilon)\right)$ time. The total preprocessing time and space over all sources is $O\left((1/\varepsilon)^6 n \log(1/\varepsilon)\right)$ and $O\left((1/\varepsilon)^4 n\right)$, respectively. Finally, if $\delta_G[s,t] \leq 1$, the shortest-path distance of s and t is their Euclidean distance. We do not know a priori which of the cases we are in, so we try all of them and return the minimum distance found.

▶ **Theorem 10** (Approximate Distance Oracle). *Given a set S of n planar points, we can construct a $(1+\varepsilon)$-approximate distance oracle for its weighted unit-disk graph with $O\left((1/\varepsilon)^3 n\log^3 n + (1/\varepsilon)^6 n\log(1/\varepsilon)\right)$ preprocessing time, $O\left((1/\varepsilon)n\log^2 n + (1/\varepsilon)^4 n\right)$ space and $O\left((1/\varepsilon)\log^2 n\right)$ query time.*

To reduce the query time, we can combine the above method with Gao and Zhang's WSPD-based oracle [13, Section 5.1], which requires $O(1)$ query time and $O((1/\varepsilon)n\log n)$ space. Its construction time is dominated by finding $(1+\varepsilon)$-approximate shortest-path distances for $O\left((1/\varepsilon)^4 n\log n\right)$ pairs, however, we can compute these distances by querying our oracle of Theorem 10 in $O\left((1/\varepsilon)^4 n\log n\cdot(1/\varepsilon)\log^2 n\right)$ total time.

▶ **Corollary 11** (Approximate Distance Oracle with $O(1)$ Query Time). *Given a set S of n planar points, we can construct a $(1+\varepsilon)$-approximate distance oracle for its weighted unit-disk graph of $O\left((1/\varepsilon)^5 n\log^3 n + (1/\varepsilon)^6 n\log(1/\varepsilon)\right)$ preprocessing time, $O\left((1/\varepsilon)^4 n\log n\right)$ space, and $O(1)$ query time.*

Similarly, we can use the distance oracle of Theorem 10 to improve Gao and Zhang's results for other distance-related problems on weighted unit-disk graphs:

▶ **Corollary 12** (Approximate Radius and Bichromatic Closest Pair). *Given a set S of n planar points, we can compute, in $O\left((1/\varepsilon)^5 n\log^3 n + (1/\varepsilon)^6 n\log(1/\varepsilon)\right)$ time, a $(1+\varepsilon)$-approximation of the radius of the weighted unit-disk graph of S or of the bichromatic closest pair distance of two given subsets $A, B \subseteq S$ therein.*

▶ Remark.

- For the sake of simplicity, we did not optimize the $\mathrm{poly}(1/\varepsilon, \log n)$ factors.
- Our distance oracle in Theorem 10 can be easily modified to report an approximate shortest path, not just its distance, in additional time proportional to the number of edges in the path: every time we find approximate shortest distances in a subgraph from a portal, we also store its approximate shortest path tree.
- The same approach gives $(1+O(\varepsilon))$-approximation results for *unweighted* unit-disk graphs, assuming that the diameter and the distances of the query vertices exceed $\Omega(1/\varepsilon)$. Specifically, Lemma 7 can be modified for the unweighted case, but the error now has an extra additive term of $4+O(\varepsilon)$ [13, Lemma 6.2], which can be ignored under our assumption. Also, we need to replace the SSSP algorithm of Lemma 2 with the $O(n\log n)$-time exact SSSP algorithm by Cabello and Jejčič [6] or by Chan and Skrepetos [9].

3 Approximate apBLSP

In this section, we study the $(1+\varepsilon)$-approximate apBSLP problem. Given a set S of n planar points, let G be its complete weighted Euclidean graph, let $w_1, w_2, \ldots, w_N$, where $N = \binom{n}{2}$, be the weights of the edges of G in non-decreasing order, and let G^i be the subgraph of G that contains only the edges of weight at most w_i. We can assume that $w_1 \geq 1$; else, we can impose that assumption by simply translating and rescaling S in linear time. We want to preprocess S into a data structure, such that we can quickly answer $(1+\varepsilon)$-approximate *bounded-leg distance* queries, i.e., given $s, t \in S$ and a positive number L, compute a $(1+\varepsilon)$-approximation of $d_{G^i}[s,t]$, where i is the largest integer with $w_i \leq L$. First, we briefly review the previous methods of Bose et al. [2] and of Roditty and Segal [24], in Section 3.1, and then describe our own approach, in Section 3.2.

3.1 Previous methods

Let $s, t \in S$, and let $c(s,t)$ be the minimum index, such that s and t are connected in $G^{c(s,t)}$. Then, since each G^i is a subgraph of G^{i+1}, we have that $d_G[s,t] \leq d_{G^{N-1}}[s,t] \leq \cdots \leq d_{G^{c(s,t)}}[s,t]$. Moreover, the s-to-t shortest path in any G^i with $i > c(s,t)$ must have an edge of weight at least $w_{c(s,t)}$, so $d_G[s,t] \geq w_{c(s,t)}$; any shortest path has at most $n-1$ edges, thus $d_{G^{c(s,t)}}[s,t] \leq (n-1)d_G[s,t]$. Therefore, as Roditty and Segal [24, Section 2] noticed, we can compute and store, for each $s, t \in S$, a $(1+\varepsilon)$-approximation of the s-to-t shortest path distance in only $O\left(\log_{1+\varepsilon} n\right)$ graphs, such that a bounded-leg distance query can be answered with a binary search in $O\left(\log\log_{1+\varepsilon} n\right)$ time.

Specifically, for every $s, t \in S$ and $j \in \{0, 1, \ldots, \lceil \log_{1+\varepsilon} n \rceil\}$, let $I^j(s,t)$ be the set of indices of the graphs G^i, such that $(1+\varepsilon)^j \delta_G[s,t] \leq \delta_{G^i}[s,t] \leq (1+\varepsilon)^{j+1}\delta_G[s,t]$. If $I^j(s,t) \neq \emptyset$, we create two values $m^j(s,t)$ and $\ell^j(s,t)$, where the former is any index therein and the latter is equal to $w_{m^j(s,t)}$; else, $m^j(s,t)$ and $\ell^j(s,t)$ are undefined. The total space required over all pairs of S is $O\left(n^2 \log_{1+\varepsilon} n\right)$. Then, given a positive number L, we can find the largest i among the $m^j(s,t)$'s, such that $w_i \leq L$, with a binary search over the $\ell^j(s,t)$'s, in $O\left(\log\log_{1+\varepsilon} n\right)$ time, and return a $(1+\varepsilon)$-approximation of the s-to-t shortest path distance in G^i.

To compute a possible index for $m^j(s,t)$, for every $s, t \in S$ and $j \in \{0, 1, \ldots, \lceil \log_{1+\varepsilon} n \rceil\}$, Roditty and Segal performed $O\left(n^2 \log_{1+\varepsilon} n\right)$ independent binary searches, each making $O(\log n)$ $(1+\varepsilon)$-approximate bounded-leg distance queries (i.e., a query to find a $(1+\varepsilon)$-approximation of the s-to-t shortest path distance in some graph G^i). Instead, we group the queries for all s, t, j into $O\left(\log n \cdot \log_{1+\varepsilon} n\right)$ rounds of n^2 offline queries each, where "offline" means that the queries in every round are given in advance.

▶ **Lemma 13** (Framework for Approximate apBLSP). *Given a set S of n planar points, we can construct a data structure for the $(1+\varepsilon)$-approximate apBLSP problem of $O\left((1/\varepsilon)n^2 \log n\right)$ space, $O(\log\log n + \log(1/\varepsilon))$ query, and $O\left(T_{\text{offline}}(n, n^2, 1+\varepsilon) \cdot (1/\varepsilon)\log^2 n\right)$ preprocessing time, where $T_{\text{offline}}(n', q', 1+\varepsilon')$ denotes the total time for answering q offline $(1+\varepsilon')$-approximate bounded-leg distance queries for an n'-point set.*

To address each round, Roddity and Segal's method would imply constructing in near-linear time a sparse $(1+\varepsilon)$-spanner of every graph G^i and then running Dijkstra's algorithm therein to answer each query; thus a near-cubic bound would be obtained for $T_{\text{offline}}(n, n^2, 1+\varepsilon)$. Instead, we show that by employing our $(1+\varepsilon)$-approximate distance oracle of Corollary 11 for weighted unit-disk graphs as a subroutine, we can obtain a truly subcubic bound on $T_{\text{offline}}(n, n^2, 1+\varepsilon)$, as we next describe.

3.2 Improved method

We view the problem of answering, for each $s, t \in S$ and $j \in \{0, 1, \ldots, \lceil \log_{1+\varepsilon} n \rceil\}$, n^2 approximate offline bounded-leg distance queries as the problem of constructing and querying the following offline *semi-dynamic* (actually insertion-only) distance oracle.

▶ **Subproblem 1** (Semi-Dynamic Approximate Distance Oracles). *Given an arbitrary graph of n vertices with edge weights in $[1, \infty)$, we want to perform an offline sequence of q operations, each of which is either an edge insertion, or a query to compute a $(1+\varepsilon)$-approximation of the shortest-path distance between two vertices. Let $T_{\text{dyn}}(n, q, 1+\varepsilon)$ be the complexity of this problem.*

We could reduce our problem to Subproblem 1 by naively inserting the $O\left(n^2\right)$ edges of G in increasing order of weight to an initially empty graph and mix that sequence of

insertions with the given sequence of bounded-leg distance queries. Hence, we would have that $T_{\text{offline}}(n, n^2, 1+\varepsilon) = O\left(T_{\text{dyn}}(n, n^2, 1+\varepsilon)\right)$.

Instead, we propose a better reduction that employs a simple periodic rebuilding trick. First, we divide the sequence of the q edge insertions and queries into $O(q/r)$ *phases* of at most r operations each, where r is a parameter to be set later. At the beginning of each phase, the current graph is a weighted unit-disk graph (after rescaling), so we can build the $(1+\varepsilon)$-approximate distance oracle of Corollary 11 in $O\left((1/\varepsilon)^5 n \log^3 n\right)$ time. Then, in $O\left(r^2\right)$ total time, we query that oracle to approximate the shortest-path distances between all pairs of vertices that are involved in the upcoming r operations (i.e., are endpoints of the edges to be inserted, or belong to the pairs to be queried). We build the complete graph over these at most $2r$ vertices, with the approximate shortest-path distances as edge weights. Each phase can then be handled by r edge insertions/queries on this smaller graph in $O\left(T_{\text{dyn}}(2r, r, 1+\varepsilon)\right)$ time. The resulting approximation factor is at most $(1+\varepsilon)^2 = 1+\Theta(\varepsilon)$. Thus, for $q = n^2$, we get the following bound:

$$T_{\text{offline}}(n, n^2, 1+\Theta(\varepsilon)) \;=\; O\left(\frac{n^2}{r} \cdot \left((1/\varepsilon)^5 n \log^3 n + r^2 + T_{\text{dyn}}(2r, r, 1+\varepsilon)\right)\right). \tag{1}$$

To solve Subproblem 1, we could do nothing during insertions and, in each query, re-run Dijkstra's algorithm from scratch. Then we would have that $T_{\text{dyn}}(2r, r, 1+\varepsilon) = O(r^3)$ and, by setting $r = (1/\varepsilon)^{5/3} n^{1/3} \log n$, $T_{\text{offline}}(n, n^2, 1+\Theta(\varepsilon))$ would be truly subcubic, namely $O\left((1/\varepsilon)^{10/3} n^{8/3} \log^2 n\right)$.

Actually, by using fast matrix multiplication and additional techniques, we can establish a better bound on $T_{\text{offline}}(n, n^2, 1+\Theta(\varepsilon))$. Our idea is to recursively divide phases into subphases, as in the proof of the following lemma. Note that this lemma actually holds for general graphs (although (semi-)dynamic shortest paths have been extensively studied in the literature, we are unable to find a known specific result that we can directly invoke).

▶ **Lemma 14** (A Semi-Dynamic Approximate Distance Oracle)**.** *We can solve Subproblem 1 in $T_{\text{dyn}}(2r, r, 1+\Theta(\varepsilon)) = O\left((1/\varepsilon) r^\omega \log r \log \overline{W}\right)$ total time, where ω is the matrix multiplication exponent and $\overline{W}$ is an upper bound on the maximum (finite) shortest-path distances.*

Proof. Let H be the input graph of $2r$ vertices, and let H' be the graph that results from performing to H all edge insertions of the first $r/2$ operations. We run the $O\left((1/\varepsilon) r^\omega \log \overline{W}\right)$-time $(1+\varepsilon)$-approximate APSP algorithm of Zwick [29] on H and H' and answer all distance queries therein. Then, we construct two graphs H_1 and H_2 of r vertices each, where H_1 (resp. H_2) is the complete graph over all vertices that are involved in the first (resp. last) $r/2$ operations; we set each edge weight in H_1 (resp. H_2) to be a $(1+\varepsilon)$-approximation of the shortest-path distance of its endpoints in H (resp. H') (which we have already computed), increasing thus the error by a $1+\varepsilon$ factor. Finally, we recurse in H_1 and H_2.

The running time of our approach is $T_{\text{dyn}}(2r, r, (1+\varepsilon)^i) \leq 2T_{\text{dyn}}(r, r/2, (1+\varepsilon)^{i+1}) + O\left((1/\varepsilon) r^\omega \log \overline{W}\right)$, where, initially, and $i = 1$; thus, we have that $T_{\text{dyn}}(2r, r, (1+\varepsilon)) = O\left((1/\varepsilon) r^\omega \log r \log \overline{W}\right)$. The approximation factor is $(1+\varepsilon)^{\log r} = 1+\Theta(\varepsilon \log r)$, which can be refined to $1+\varepsilon$, by resetting $\varepsilon \leftarrow \varepsilon/\log r$. ◀

Combining (1) with the above lemma gives

$$T_{\text{offline}}(n, n^2, 1+\Theta(\varepsilon)) \;=\; O\left(\frac{n^2}{r}\left((1/\varepsilon)^5 n \log^3 n + (1/\varepsilon) r^\omega \log r \log \overline{W}\right)\right).$$

Setting $r = n^{1/\omega}$ yields $T_{\text{offline}}(n, n^2, 1+\Theta(\varepsilon)) = O\left((1/\varepsilon)^5 n^{3-1/\omega} \log^3(n\overline{W})\right)$, where $\overline{W} \leq nW$, and W is the spread of S.

▶ **Theorem 15** (Approximate apBLSP). *Given a set S of n planar points of spread W, we can construct a data structure for the $(1+\varepsilon)$-approximate apBLSP problem of $O\left((1/\varepsilon)n^2 \log n\right)$ space, $O\left(\log\log n + \log(1/\varepsilon)\right)$ query, and $O\left((1/\varepsilon)^6 n^{3-1/\omega} \log^5(nW)\right)$ preprocessing time.*

The current best bound on the matrix multiplication exponent [26, 20] is $\omega < 2.373$, which gives a preprocessing time of $O\left((1/\varepsilon)^6 n^{2.579} \log^5(nW)\right)$.

Remark. For the sake of simplicity, we did not optimize the $\text{poly}(1/\varepsilon, \log(nW))$ factors. Specifically, it might be possible to avoid the dependency on the spread W by using known techniques, such as balanced quadtrees.

References

1 Srinivasa Arikati, Danny Z. Chen, L. Paul Chew, Gautam Das, Michiel Smid, and Christos D. Zaroliagis. Planar spanners and approximate shortest path queries among obstacles in the plane. In *Proceedings of the 4th Annual European Symposium on Algorithms (ESA)*, pages 514–528, 1996.

2 Prosenjit Bose, Anil Maheshwari, Giri Narasimhan, Michiel Smid, and Norbert Zeh. Approximating geometric bottleneck shortest paths. *Computational Geometry*, 29(3):233–249, 2004.

3 Costas Busch, Ryan LaFortune, and Srikanta Tirthapura. Improved sparse covers for graphs excluding a fixed minor. In *Proceedings of the 26th Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 61–70, 2007.

4 Sergio Cabello. Many distances in planar graphs. *Algorithmica*, 62(1-2):361–381, 2012. doi:10.1007/s00453-010-9459-0.

5 Sergio Cabello. Subquadratic algorithms for the diameter and the sum of pairwise distances in planar graphs. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2143–2152, 2017.

6 Sergio Cabello and Miha Jejčič. Shortest paths in intersection graphs of unit disks. *Computational Geometry*, 48(4):360–367, 2015.

7 Paul B. Callahan and S. Rao Kosaraju. A decomposition of multidimensional point sets with applications to k-nearest-neighbors and n-body potential fields. *Journal of the ACM*, 42(1):67–90, 1995. doi:10.1145/200836.200853.

8 Timothy M. Chan. A dynamic data structure for 3-d convex hulls and 2-d nearest neighbor queries. *Journal of the ACM*, 57(3):16:1–16:15, 2010. doi:10.1145/1706591.1706596.

9 Timothy M. Chan and Dimitrios Skrepetos. All-pairs shortest paths in unit-disk graphs in slightly subquadratic time. In *Proceedings of the 27th Annual International Symposium on Algorithms and Computation (ISAAC)*, pages 24:1–24:13, 2016.

10 Timothy M. Chan and Dimitrios Skrepetos. Faster approximate diameter and distance oracles in planar graphs. In *Proceedings of the 25th European Symposium on Algorithms (ESA)*, pages 25:1–25:13, 2017.

11 Vincent Cohen-Addad, Søren Dahlgaard, and Christian Wulff-Nilsen. Fast and compact exact distance oracle for planar graphs. In *Proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 962–973, 2017. doi:10.1109/FOCS.2017.93.

12 David Eppstein, Gary L. Miller, and Shang-Hua Teng. A deterministic linear time algorithm for geometric separators and its applications. *Fundamenta Informaticae*, 22(4):309–329, 1995.

13 Jie Gao and Li Zhang. Well-separated pair decomposition for the unit-disk graph metric and its applications. *SIAM Journal on Computing*, 35(1):151–169, 2005. Preliminary version in STOC 2003.

14 Pawel Gawrychowski, Shay Mozes, Oren Weimann, and Christian Wulff-Nilsen. Better tradeoffs for exact distance oracles in planar graphs. In *Proceedings of the 29th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 515–529, 2018.

15 Pawel, Gawrychowski, Haim Kaplan, Shay Mozes, Micha Sharir, and Oren Weimann. Voronoi diagrams on planar graphs, and computing the diameter in deterministic $\tilde{O}(n^{5/3})$ time. In *Proceedings of the 29th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 495–514, 2018.

16 Qian-Ping Gu and Gengchun Xu. Constant query time $(1+\varepsilon)$-approximate distance oracle for planar graphs. In *Proceedings of the 26th International Symposium on Algorithms and Computation (ISAAC)*, pages 625–636, 2015.

17 Haim Kaplan, Wolfgang Mulzer, Liam Roditty, Paul Seiferth, and Micha Sharir. Dynamic planar Voronoi diagrams for general distance functions and their algorithmic applications. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2495–2504, 2017.

18 Ken-ichi Kawarabayashi, Christian Sommer, and Mikkel Thorup. More compact oracles for approximate distances in undirected planar graphs. In *Proceedings of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 550–563, 2013.

19 Philip Klein. Preprocessing an undirected planar network to enable fast approximate distance queries. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 820–827, 2002.

20 François Le Gall. Powers of tensors and fast matrix multiplication. In *Proceedings of the 39th International Symposium on Symbolic and Algebraic Computation (ISSAC)*, pages 296–303, 2014. doi:10.1145/2608628.2608664.

21 Xiang-Yang Li, Gruia Calinescu, and Peng-Jun Wan. Distributed construction of a planar spanner and routing for ad hoc wireless networks. In *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, volume 3, pages 1268–1277, 2002.

22 Gary L Miller, S-H Teng, and Stephen A Vavasis. A unified geometric approach to graph separators. In *Proceedings of the 32nd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 538–547, 1991.

23 Franco P. Preparata and Michael Ian Shamos. *Computational Geometry: An Introduction*. Springer-Verlag, 1985.

24 Liam Roditty and Michael Segal. On bounded leg shortest paths problems. *Algorithmica*, 59(4):583–600, 2011. Preliminary version in SODA 2007.

25 Mikkel Thorup. Compact oracles for reachability and approximate distances in planar digraphs. *Journal of the ACM*, 51(6):993–1024, 2004.

26 Virginia Vassilevska Williams. Multiplying matrices faster than Coppersmith–Winograd. In *Proceedings of the 44th ACM Symposium on Theory of Computing (STOC)*, pages 887–898, 2012. doi:10.1145/2213977.2214056.

27 Oren Weimann and Raphael Yuster. Approximating the diameter of planar graphs in near linear time. *ACM Transactions on Algorithms*, 12(1):1–13, 2016.

28 Chenyu Yan, Yang Xiang, and Feodor F. Dragan. Compact and low delay routing labeling scheme for unit disk graphs. *Computational Geometry*, 45(7):305–325, 2012. doi:10.1016/j.comgeo.2012.01.015.

29 Uri Zwick. All pairs shortest paths using bridging sets and rectangular matrix multiplication. *Journal of the ACM*, 49(3):289–317, 2002.