SIAM J. Comput.
Vol. 36, No. 3, pp. 815–834

COMPUTING MAXIMALLY SEPARATED SETS IN THE PLANE*

PANKAJ K. AGARWAL†, MARK OVERMARS‡, AND MICHA SHARIR§

Abstract. Let S be a set of n points in $\mathbb{R}^2$. Given an integer $1 \le k \le n$, we wish to find a *maximally separated subset* $I \subseteq S$ of size k; this is a subset for which the minimum among the $\binom{k}{2}$ pairwise distances between its points is as large as possible. The decision problem associated with this problem is to determine whether there exists $I \subseteq S$, $|I| = k$, so that all $\binom{k}{2}$ pairwise distances in I are at least 2. This problem can also be formulated in terms of disk-intersection graphs: Let D be the set of unit disks centered at the points of S. The *disk-intersection* graph G of D has as edges all pairs of disks with nonempty intersection. Any set I with the above properties is then the set of centers of disks that form an independent set in the graph G. This problem is known to be NP-complete if k is part of the input. In this paper we first present a linear-time ε-approximation algorithm for any constant k. Next we give exact algorithms for the cases $k = 3$ and $k = 4$ that run in time $O(n^{4/3}\text{polylog}(n))$. We also present a simpler $n^{O(\sqrt{k})}$-time exact algorithm (as compared with the recent algorithm in [J. Alber and J. Fiala, *J. Algorithms*, 52 (2004), pp. 134–151]) for arbitrary values of k.

Key words. disk-intersection graphs, independent set, geometric optimization

AMS subject classifications. 68Q25, 68U05, 68W25

DOI. 10.1137/S0097539704446591

1. Introduction. Let S be a set of n points in the plane. We are interested in finding a small subset I of S such that all the pairwise distances between points in I are large. To be more precise, let I be a subset of S of cardinality k, for $1 \le k \le n$. We define the *separation distance* $d_{\text{sep}}(I)$ to be the minimum among the $\binom{k}{2}$ pairwise distances between its k points. We call I *δ-separated* if $d_{\text{sep}}(I) \ge \delta$. We call I a *maximally separated* subset of S if $d_{\text{sep}}(I) \ge d_{\text{sep}}(I')$ for all subsets $I' \subseteq S$ of size k. Let $d^k_{\text{sep}}(S) = \max_{I \subseteq S,\ |I|=k} d_{\text{sep}}(I)$.

In this paper we study algorithms for computing such maximally separated subsets. We consider small (constant) values of k, but we also address the general case. For the case $k = 2$ the problem is equivalent to finding a diametral pair of S and thus can be solved (exactly) in $O(n \log n)$ time [10], and can be ε-approximated in linear time (see, e.g., [1]). For larger k, the problem becomes considerably more complicated, and is known to be NP-complete if k is part of the input [9].

---

*Received by the editors November 29, 2004; accepted for publication (in revised form) April 6, 2006; published electronically November 3, 2006. The work of P.A. and M.S. was supported by a grant from the U.S.-Israeli Binational Science Foundation. Work by P.A. was also supported by NSF under grants CCR-00-86013, EIA-98-70724, EIA-99-72879, EIA-01-31905, and CCR-02-04118. Work by M.S. was also supported by NSF grants CCR-97-32101 and CCR-00-98246, by a grant from the Israel Science Fund (for a Center of Excellence in Geometric Computing), and by the Hermann Minkowski–MINERVA Center for Geometry at Tel Aviv University. Part of the research was done during the 2003 Bellairs Workshop on Computational Geometry, organized by Godfried Toussaint, and the 2003 Dagstuhl Workshop on Computational Geometry. A preliminary version of this paper has appeared in *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms*, New Orleans, 2004, pp. 509–518.

http://www.siam.org/journals/sicomp/36-3/44659.html

†Department of Computer Science, Duke University, Durham, NC 27708-0129 (pankaj@cs.duke.edu).

‡Department of Computer Science, Utrecht University, Utrecht, The Netherlands (markov@cs.uu.nl).

§School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel, and Courant Institute of Mathematical Sciences, New York University, New York, NY 10012 (michas@post.tau.ac.il).

Finding small well-separated subsets is important in certain pattern-matching problems, where the points in the subset form a representation of the total set of points. For example, Vleugels and Veltkamp [23] describe a method for fast indexing of multimedia databases using so-called *vantage objects*. These vantage objects are points in the feature space for the matching problem. It has been observed that, for the application at hand, the chosen vantage objects best be well-separated.

The decision problem associated with the problem of computing a maximally separated subset of size k calls for determining whether a δ-separated subset I of size k exists for a given $\delta > 0$. This problem can also be formulated in terms of disk-intersection graphs: Let D be the set of disks of radius $\delta/2$ centered at the points of S. The *disk-intersection* graph G of D has the disks as nodes and two disks are connected by an edge if they intersect. Clearly, a δ-separated subset I is the set of centers of an independent set in G (and vice versa). So the decision problem is equivalent to the problem of finding an independent set of size k in the disk-intersection graph G. Recently, the problem of computing the maximum independent set in intersection graphs has attracted considerable attention because of its application in geographic information systems (GIS); see [2, 11, 12] and references therein.

Related work. The problem of computing an independent set in a graph is one of the earliest problems known to be NP-complete [13]. In fact, for a general graph with n vertices, there cannot be a polynomial-time algorithm with approximation ratio better than $n^{1-\varepsilon}$, for any $\varepsilon > 0$, unless $NP = ZPP$ [15]. The best known polynomial-time algorithm finds an independent set of size $\Omega((\kappa \log^2 n)/n)$, where κ is the size of the maximum independent set in the graph [6]. However, better approximation algorithms are known for intersection graphs of geometric objects. The maximum independent set in the intersection graph of intervals on a line can be computed in polynomial time, but the problem remains NP-complete for intersection graphs of orthogonal segments, unit disks, and unit squares [9]. Still, ε-approximation algorithms have been proposed for intersection graphs of unit disks, unit squares, arbitrary disks, and fat objects [7, 9, 11, 17, 19, 20, 22], and an $O(\log n)$-approximation algorithm is known for intersection graphs of rectangles [2].

Little is known about computing maximally separated sets. Formann and Wagner [12] developed a 2-approximation algorithm under the L_∞-metric. Alber and Fiala [5] present an algorithm that computes, in time $n^{O(\sqrt{k})}$, an independent set of cardinality k in the intersection graph of a set of disks. Their algorithm, however, is rather complicated, and they do not consider cases involving small values of k. Moreover, since they compute all pairwise distances between input points, their algorithm takes $\Omega(n^2)$ time even for small values of k.

Our results. In this paper we consider both the general problem and special instances of it that involve small values of k, and develop exact and approximation algorithms for these problems. The paper contains four main results:

- (i) For any constant k, we present in Section 2 a simple, linear-time algorithm that returns a subset I of size k such that $d_{\text{sep}}(I) \geq (1-\varepsilon) d^k_{\text{sep}}(S)$. Such an approximation algorithm is suitable for the pattern-matching application mentioned above.
- (ii, iii) We present in Sections 3 and 4 $O(n^{4/3}\text{polylog}(n))$-time algorithms for computing (exactly) maximally separated subsets of size 3 and 4, respectively.
- (iv) We also present, in Section 5, a simpler $n^{O(\sqrt{k})}$-time exact algorithm (as compared with the algorithm in [5]) for arbitrary values of k.

2. An ε-approximation algorithm. In this section we show that, for any constant k and for any constant $0 < \varepsilon < 1$, we can find in linear time a subset I of S of cardinality k such that $d_{\text{sep}}(I) \geq (1-\varepsilon)d_{\text{sep}}^k(S)$. The running time is exponential in k. We refer to such an I as an *ε-approximation* of the optimal solution.

As a warm-up exercise let us consider the case $k = 2$. We want to find an ε-approximation of the diameter of the set S in linear time. This is an already solved problem (see, e.g., [1]), but we sketch a solution (a) for the sake of completeness, and (b) to prepare for tackling the general case $k \geq 3$.

Let B be the axis-parallel bounding box of S. Let w be the width of B and h its height, and let us assume, without loss of generality, that $w \geq h$. Clearly, the diameter d lies between w and $\sqrt{2}w$. Choose $\delta = \varepsilon w/2\sqrt{2}$ and divide the box B into $O\left(1/\varepsilon^2\right)$ squares of size $\delta \times \delta$. In each nonempty square τ we pick a single point from $\tau \cap S$ and retain only the highest and lowest point in each row or column of the grid. So we end up with a subset S' of S with $O(1/\varepsilon)$ points. We compute the diameter d' of S' exactly, which takes $O((1/\varepsilon)\log(1/\varepsilon)) = O(1)$ time. Now it is easy to see that the actual diameter d satisfies

$$d \leq d' + 2\sqrt{2}\delta.$$

So

$$d' \geq d - 2\sqrt{2}\delta \geq d - \varepsilon w \geq (1-\varepsilon)d,$$

since $d \geq w$. Hence the diameter of the set S' is an ε-approximation for the diameter of S. As computing the bounding box and the set S' takes $O(n)$ time, this procedure computes an ε-approximate diameter of S in $O(n + (1/\varepsilon)\log(1/\varepsilon))$ time.

Let us next assume that $k \geq 3$. Our algorithm uses recursion on k. As above, we compute the smallest axis-parallel bounding box B of S, denote its width and height as w and h, respectively, and assume that $w \geq h$.

We first consider the case in which $d_{\text{sep}}^k(S) \leq w/(k+1)$. (Note that this case cannot arise for $k = 2$.) We subdivide the box B into $k+1$ vertical strips $s_0, \ldots, s_k$, each of width $w/(k+1)$, and set $S_i := S \cap s_i$, for $i = 0, \ldots, k$. Any solution will use points from at most k of these $k+1$ strips. Therefore, for each strip s_i, we compute an ε-approximation of a maximally separated set in $S \setminus S_i$. The best among those $k+1$ solutions is the answer we are looking for.

For each $i = 0, \ldots, k$, we process $S \setminus S_i$ as follows. A crucial observation is that, for $1 \leq i \leq k-1$, we may assume that the optimal solution uses points that *lie on both sides* of s_i. Indeed, if an optimal solution I consists only of points that lie, say, to the right of s_i, replace I by $I' \cup \{p_L\}$, where p_L is the leftmost point of P (which lies on the left edge of B), and I' is an optimal solution with $k-1$ points for the subset S_R of S that lies to the right of s_i. Since $d_{\text{sep}}^k(S) \leq w/(k+1)$, it follows that the separation of the new solution, which does have points on both sides of s_i, is at least as large as that of $d_{\text{sep}}(I)$. The same argument works for $i = 0$ and for $i = k$ (in the latter case we use the rightmost point p_R of S instead of p_L). As is easily seen, these observations also carry over to ε-approximations of the optimal solution.

Hence, for $i = 0$ and $i = k$, we invoke the procedure recursively for finding an ε-approximate solution with $k-1$ points for the set $S \setminus S_i$, and then add to the solution the leftmost point of S (for $i = 0$) or the rightmost point (for $i = k$).

Consider then the case $1 \leq i \leq k-1$. Put $S_L := \bigcup_{j<i} S_j$ and $S_R := \bigcup_{j>i} S_j$. We need to guess the number t of points of the optimal solution that lie in S_L (so that $k-t$ points lie in S_R). As argued above, we may assume that $1 \leq t \leq k-1$. For each value of t in this range, we compute recursively an ε-approximation I_L (resp., I_R) of a maximally separated set of size t in S_L (resp., of size $k-t$ in S_R). Then $I_L \cup I_R$ form an ε-approximation of the optimal solution to the whole problem. Thus, for each strip we solve $2(k-1)$ problems with size smaller than k. In total, we need to solve $O(k^2)$ subproblems. Denoting by $T_k(n, \varepsilon)$ the maximum time needed to ε-approximate $d^k_{\text{sep}}(S)$, over sets S of n points, we thus obtain a procedure that handles the case $d^k_{\text{sep}}(S) \le w/(k+1)$, with total cost of $O(n + k^2 T_{k-1}(n, \varepsilon))$.

So we are left with the case in which the maximal separation distance $d^k_{\text{sep}}(S)$ is larger than $w/(k+1)$. We proceed in a manner similar to that for the case $k = 2$. Let

$$\delta = \frac{\varepsilon w}{2\sqrt{2}(k+1)}.$$

We partition the bounding box B of the set S into $O(k^2/\varepsilon^2)$ grid cells of size at most $\delta \times \delta$, choose an arbitrary single point of S from each nonempty cell of the grid, obtain a set A of $O(k^2/\varepsilon^2)$ representative points, and compute an exact maximally separated set I of size k for A, using any, potentially brute-force, method. (One possibility is to use the algorithm presented in Section 5.)

We claim that $d_{\text{sep}}(I) \ge (1-\varepsilon) d^k_{\text{sep}}(S)$. Indeed, let $\{p_1, \ldots, p_k\} \subseteq S$ be a maximally separated set of S of size k. Since $\varepsilon < 1$, these points must lie in different cells. Let $p'_i \in A$ be the representative point from the cell in which p_i lies, and let $I' = \{p'_1, \ldots, p'_k\}$. As in the case $k = 2$, it is easily seen that

$$d_{\text{sep}}(I') \ge d^k_{\text{sep}}(S) - 2\sqrt{2}\delta = d^k_{\text{sep}}(S) - \frac{\varepsilon w}{k+1} > (1-\varepsilon) d^k_{\text{sep}}(S),$$

because $d^k_{\text{sep}}(S) > w/(k+1)$. Since we solve the problem exactly for A, $d_{\text{sep}}(I) \ge d_{\text{sep}}(I') > (1-\varepsilon) d^k_{\text{sep}}(S)$, as asserted. The running time bound $T_k(n, \varepsilon)$ thus satisfies the recurrence

$$T_k(n, \varepsilon) = O(n + k^2 T_{k-1}(n, \varepsilon) + C_k(O(k^2/\varepsilon^2))),$$

where $C_k(m)$ is the time needed to compute exactly a maximally separated subset of size k in a set of m points. Clearly, the solution of this recurrence is $O(n)$, for any constant k. More precisely, it is upper bounded by $c^k (k!)^2 n + b(k)/\varepsilon^{a(k)}$, for some constant $c > 0$, where $b(k)$ is exponential in k, and $a(k)$ is at most linear in k. (For example, using a brute-force solution for the case of large separation, for which $C_k(m) = O(m^k)$, we obtain $b(k) \le (c'k)^{2k}$, for some constant $c' > 0$, and $a(k) = 2k$.) Thus, we have the following theorem.

Theorem 2.1. *For a set S of n points in $\mathbb{R}^2$ and any constants k and $0 < \varepsilon < 1$, we can compute in $k^{O(k)} n + (k/\varepsilon)^{O(k)}$ time a subset $I \subseteq S$ of size k such that $d_{\text{sep}}(I) \ge (1-\varepsilon) d^k_{\text{sep}}(S)$.*

3. Computing a maximally separated triple. Let S be a set of n points in $\mathbb{R}^2$. We wish to compute a maximally separated triple in S. Our overall approach consists of three steps. First, we perform a binary search on the pairwise distances of S, and for each distance δ that the search encounters, we determine whether S contains a δ-separated triple. Next, in order to determine the existence of a δ-separated triple, we draw a sufficiently small grid within the bounding box of S so that each point of a δ-separated triple of S lies in a distinct grid cell. We thus reduce the problem of computing a δ-separated triple to a trichromatic variant of this problem. Finally, we determine the existence of a trichomatic δ-separated triple in $O(n^{4/3} \log^2 n)$ time.

For simplicity, we describe these steps in the reverse order. That is, we first describe the decision algorithm for the trichromatic version, then we show how to reduce the original decision problem to the trichromatic problem, and finally we sketch the binary-search procedure.

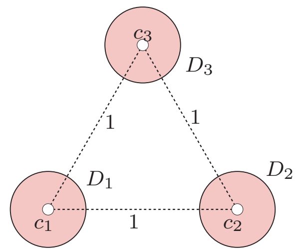

FIG. 1. *An instance of three point sets (contained in the shaded disks) with property* ($\triangle$).

We need a few notations. First, we may assume for the decision problem that $\delta = 1$. For a point $p \in \mathbb{R}^2$, let $\mathbb{D}(p)$ denote the disk of unit radius centered at p. For a set A of points in $\mathbb{R}^2$, let $K(A) = \bigcap_{p\in A} \mathbb{D}(p)$. $K(A)$ is a convex region bounded by circular arcs that lie on the boundaries of the disks $\mathbb{D}(p)$, and each disk contributes at most one such arc to $\partial K(A)$; $K(A)$ can be constructed in time $O(|A| \log |A|)$ [10].

3.1. Computing a trichromatic 1-separated triple. Let S_1, S_2, and S_3 be three sets of n points each in $\mathbb{R}^2$ that satisfy the following property.

($\triangle$) There is a constant $\delta \le 1/6$ so that, for $i = 1, 2, 3$, S_i is contained in a disk D_i of radius δ centered at a point c_i, and $|c_1c_2| = |c_2c_3| = |c_3c_1| = 1$.

Without loss of generality, we assume that $c_1 = (0, 0), c_2 = (1, 0)$, and $c_3 = (1/2, \sqrt{3}/2)$; see Figure 1. We wish to compute a 1-separated triple in $S_1 \times S_2 \times S_3$, or to determine that no such triple exists. Clearly, no other triple of points in $S_1 \cup S_2 \cup S_3$ can be 1-separated.

Before continuing, we remark that, informally, property ($\triangle$) captures the hard case for finding a 1-separated trichromatic triple. If two of the sets S_i are too close to each other, then no trichromatic 1-separated triple exists, and if some pairs of sets are too far apart, the problem reduces to finding a diametral pair. This will be discussed in detail in section 3.2.

Let $G \subseteq S_1 \times S_2$ denote the bipartite graph

$$G = \{(p, q) \mid p \in S_1, q \in S_2;\ |pq| \ge 1\}.$$

Using the algorithm of Katz and Sharir [18], we compute, in $O(n^{4/3} \log n)$ time, a family $\mathcal{F} = \{A_1 \times B_1, \ldots , A_u \times B_u\}$, which is a partition of G into complete bipartite graphs, satisfying

$$\sum_i (|A_i| + |B_i|) = O(n^{4/3} \log n).$$

For each $1 \le i \le u$, let $R_i = K(A_i) \cup K(B_i)$. Set $\mathcal{R} := \bigcap_{i=1}^{u} R_i$. The following lemma is a straightforward reformulation of the original problem.

LEMMA 3.1. *There exists a* 1*-separated (trichromatic) triple in* $S_1 \times S_2 \times S_3$ *if and only if* $S_3 \not\subseteq \mathcal{R}$.

Proof. Let $p \in S_3$ be a point that does not lie in $\mathcal{R}$. Then there exists an $i \le u$ so that $p \notin R_i = K(A_i) \cup K(B_i)$. Since $p \notin K(A_i)$, there is a point $q \in A_i$ so that

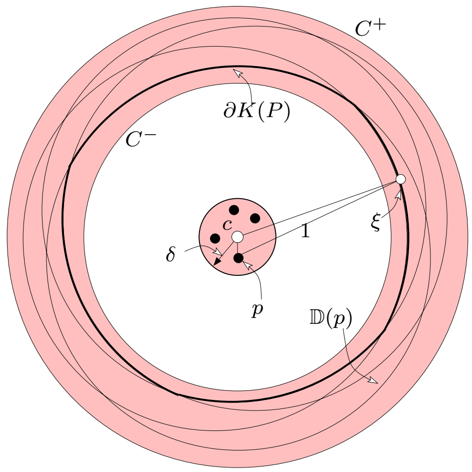

FIG. 2. *The annulus that contains $\partial K(P)$ (drawn as a thick curve).*

$p \notin \mathbb{D}(q)$. Similarly there is a point $r \in B_i$ so that $p \notin \mathbb{D}(r)$. Hence, $|pq|, |pr| \geq 1$. Moreover, $|qr| \geq 1$ because $(q, r) \in A_i \times B_i$, thereby implying that (q, r, p) is a 1-separated triple. The converse implication is established in a similar manner: Suppose that $q \in S_1$, $r \in S_2$, and $p \in S_3$ form a 1-separated triple. Since $|qr| \geq 1$, there exists an i such that $(q, r) \in A_i \times B_i$. Since $|pq|$, $|pr| \geq 1$, it follows that $p \notin K(A_i)$ and $p \notin K(B_i)$, and therefore $p \notin \mathcal{R}$. ∎

The following simple technical observation is important for our algorithm.

LEMMA 3.2. *Let P be a set of points in $\mathbb{R}^2$ lying in a disk of radius δ centered at a point c. Then $\partial K(P)$ lies between two concentric circles of radius $1+\delta$ and $1-\delta$ centered at c.*

Proof. Let C^+ (resp., C^-) denote the circle of radius $1+\delta$ (resp., $1-\delta$) centered at c. Fix a point $p \in P$. For any point $\xi \in \partial\mathbb{D}(p)$,

$$1 - \delta \leq |\xi p| - |pc| \leq |\xi c| \leq |\xi p| + |pc| \leq 1 + \delta.$$

Hence, $\partial\mathbb{D}(p)$ lies between C^- and C^+. Since this is true for every point $p \in P$, $\partial K(P)$ lies between C^- and C^+; see Figure 2. ∎

LEMMA 3.3. *For each $1 \leq i \leq u$, the upper (resp., lower) boundaries of $K(A_i)$ and $K(B_i)$ cross at exactly one point.*

Proof. Let W_1 (resp., W_2) denote the annulus bounded by the concentric circles of radii $1+\delta$ and $1-\delta$ centered at c_1 (resp., c_2). By Lemma 3.2, $\partial K(A_i)$ (resp., $\partial K(B_i)$) is contained in W_1 (resp., W_2). Therefore $\partial K(A_i) \cap \partial K(B_i) \subseteq W_1 \cap W_2$. Since $\delta < 1/6$ and $|c_1 c_2| = 1$, the inner circles of W_1 and W_2 intersect, and thus $W_1 \cap W_2$ consists of two connected components Σ^+, Σ^-, where Σ^+ lies above the x-axis and Σ^- below the x-axis; see Figure 3. An easy calculation shows that the x-coordinate of the leftmost (resp., rightmost) point of Σ^+ is $1/2 - 2\delta$ (resp., $1/2 + 2\delta$), and that the y-coordinate of the bottommost point is $\sqrt{(1-\delta)^2 - 1/4}$. Since $\delta \leq 1/6$, Σ^+ lies fully to the right of D_1, to the left of D_2, and above both these disks. This implies that, within Σ^+, the boundary of each $\mathbb{D}(p)$, for $p \in A_i$, is the graph of a strictly decreasing function, and thus $\partial K(A_i)$ is also the graph of a strictly decreasing function within Σ^+. By a fully symmetric argument, $\partial K(B_i)$ is the graph of a strictly increasing function within Σ^+. Moreover, $\partial K(A_i) \cap \Sigma^+$ is contained in the upper boundary of $K(A_i)$,

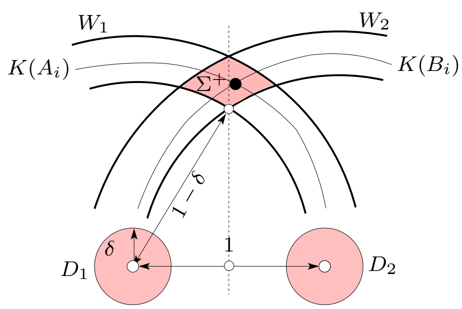

FIG. 3. *The annuli W_1, W_2 and their top intersection Σ^+. For $\delta \le 1/6$, the lowest point of Σ^+ lies above the line $y = \delta$, and its leftmost point has x-coordinate $\ge \delta$.*

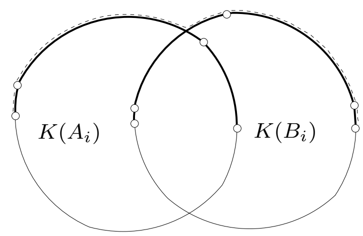

FIG. 4. $K(A_i)$, $K(B_i)$ (*whose top boundaries are drawn as thick curves*), *and the edges of* Γ_i (*drawn as dashed arcs*).

and similarly for $K(B_i)$, because Σ^+ lies above D_1 and D_2. This is easily seen to imply the assertion of the lemma. ∎

Lemma 3.3 implies that ∂R_i consists of a connected portion of $\partial K(A_i)$ and a connected portion of $\partial K(B_i)$. The leftmost and rightmost points of R_i partition ∂R_i into two parts, which we refer to as the upper and lower boundaries of R_i. Let Γ_i be the set of circular arcs forming the upper boundary of R_i; we have $|\Gamma_i| \le |A_i|+|B_i|$. See Figure 4. Set $\Gamma := \bigcup_{i=1}^{u} \Gamma_i$; then $|\Gamma| \le \sum_{i=1}^{u}(|A_i| + |B_i|)$. Let $\mathcal{L}_\Gamma$ denote the lower envelope of Γ.

LEMMA 3.4. *A point $p \in S_3$ lies inside $\mathcal{R}$ if and only if p lies below the lower envelope $\mathcal{L}_\Gamma$.*

Proof. If $p \in \mathcal{R}$, then it lies below the upper boundary of each R_i, thereby implying that p lies below $\mathcal{L}_\Gamma$. Conversely, suppose that p lies below $\mathcal{L}_\Gamma$. Then p lies below the upper boundary of every R_i. Let Σ^+ be the same region as in the proof of Lemma 3.3. Since $|c_1c_2| = |c_1c_3| = |c_2c_3| = 1$, and $\delta \le 1/6$, a simple calculation shows that $D_3 \subset \Sigma^+$, and thus S_3 is also contained in Σ^+. The argument in the proof of Lemma 3.3 implies that Σ^+ lies above the lower boundaries of every $K(A_i)$ and of every $K(B_i)$. Hence, if p lies below the boundary of each R_i, it lies in each R_i and thus also in $\mathcal{R}$. ∎

In view of Lemma 3.4, we may proceed as follows. For each i, we compute $K(A_i)$, $K(B_i)$, R_i, and Γ_i. The total time spent in this step is

$$O\left(\sum_{i=1}^{u}(|A_i| + |B_i|)\log n\right) = O\big(n^{4/3}\log^2 n\big).$$

Since each arc in Γ is a portion of the upper boundary of a unit-radius disk, two arcs of Γ intersect in at most one point. Hence, we can compute the lower envelope

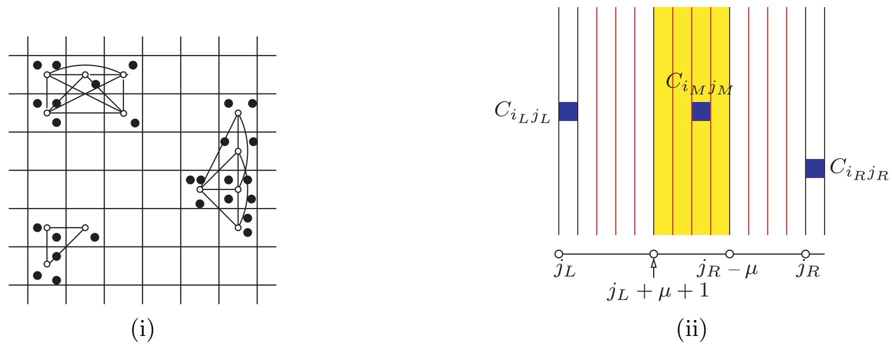

FIG. 5. (i) *Drawing a grid and the graph* $\mathcal{G}$. (ii) *The case where* $\mathcal{C}$ *spans more than* $3\mu + 1$ *columns.*

$\mathcal{L}_\Gamma$ of Γ in $O(|\Gamma| \log n)$ time, using the algorithm of Hershberger [16] (see also [21]). For each edge ξ of $\mathcal{L}_\Gamma$ we store the index j such that ξ is (a portion of) an arc in Γ_j. Finally, for each point $p \in S_3$ we determine whether p lies below or above $\mathcal{L}_\Gamma$, using a simple binary search over the arcs of $\mathcal{L}_\Gamma$. If p lies above $\mathcal{L}_\Gamma$, then the test yields an arc of $\mathcal{L}_\Gamma$ that lies below p. This arc is contained in an arc of some Γ_i, and we can thus deduce that $p \notin R_i$ (by Lemma 3.4). Then, scanning the points of $A_i \cup B_i$ in additional $O(|A_i| + |B_i|)$ time, we are certain to find a 1-separated triple $(a, b, p) \in A_i \times B_i \times S_3$. The total running time of the algorithm is $O(n^{4/3} \log^2 n)$. Hence, we obtain the following result.

THEOREM 3.5. *Let* S_1, S_2, *and* S_3 *be three finite point sets in* $\mathbb{R}^2$ *that satisfy property* ($\triangle$), *and put* $n_i = |S_i|$, *for* $i = 1, 2, 3$. *Then one can construct, in time* $O(n^{4/3} \log^2 n)$, *a* 1*-separated triple in* $S_1 \times S_2 \times S_3$, *if one exists, or determine that no such triple exists.*

3.2. Reduction to the trichromatic case. Let S be a set of n points in $\mathbb{R}^2$. We wish to compute a 1-separated triple in S if one exists, or else to determine that no such triple exists. We fix a small constant $\varepsilon \ll 1/16$, and set $\mu = \lceil 1/\varepsilon \rceil$. We draw a square grid of size ε in the plane. For $i, j \in \mathbb{Z}$, let C_{ij} denote the grid cell $[i\varepsilon, (i+1)\varepsilon) \times [j\varepsilon, (j+1)\varepsilon)$, and let $S_{ij} = S \cap C_{ij}$. Let $\mathcal{C}$ denote the set of nonempty grid cells (i.e., those with $S_{ij} \neq \emptyset$). We construct a graph $\mathcal{G} = (\mathcal{C}, \mathcal{E})$ where $(C, C') \in \mathcal{E}$ if $\min\{|pp'| \mid p \in C, p \in C'\} < 1$; see Figure 5 (i).

LEMMA 3.6. *If* $\mathcal{G}$ *is not connected, then we can compute a* 1*-separated triple in* S (*or determine that no such triple exists*) *in* $O(n \log n)$ *time.*

Proof. First, note that if two nonempty grid cells $C_{ij}, C_{kl} \in \mathcal{C}$ lie in different connected components of $\mathcal{G}$, then for any pair $(p, q) \in S_{ij} \times S_{kl}$, $|pq| \geq 1$. If $\mathcal{G}$ has three (or more) connected components $\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3$, then a 1-separated triple is obtained by choosing one point of S lying in a single grid cell of each of $\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3$. If $\mathcal{G}$ has at least two connected components, then let $S_1 \subseteq S$ be the subset of points lying in the grid cells of one connected component, and put $S_2 := S \setminus S_1$. We test, in $O(n \log n)$ time, whether $\max\{\text{diam}(S_1), \text{diam}(S_2)\} \geq 1$. Suppose that p, q is a diametral pair of, say, S_1 and that $|pq| \geq 1$; then we choose an arbitrary point $r \in S_2$ and return (p, q, r). By construction, this is a 1-separated triple. If $\text{diam}(S_1), \text{diam}(S_2)$ are both smaller than 1, then clearly no 1-separated triple exists. Hence, if $\mathcal{G}$ is not connected, then we can construct in time $O(n \log n)$ a 1-separated triple in S if one exists, or determine that no such triple exists. ∎

LEMMA 3.7. *If $\mathcal{G}$ is connected and $\mathcal{C}$ spans more than $3\mu+1$ columns or rows of the grid, i.e., it has cells in two columns (or rows) whose indices j_L, j_R satisfy $j_R - j_L \geq 3\mu+1$, then a 1-separated triple in S exists, and can be constructed in $O(n)$ time.*

Proof. Without loss of generality, it suffices to consider the case where $\mathcal{C}$ spans more than $3\mu+1$ columns. Let $C_{i_L j_L}$ (resp., $C_{i_R j_R}$) be a grid cell of $\mathcal{C}$ in the leftmost (resp., rightmost) column, let $p_L \in S_{i_L j_L}$, and let $p_R \in S_{i_R j_R}$. By assumption, $j_R - j_L \geq 3\mu+1$; see Figure 5 (ii). We group the columns between the j_Lth and j_Rth columns (exclusive) into three pairwise-disjoint vertical strips V_1, V_2, V_3, appearing in this left-to-right order, each of width at least $\mu\varepsilon \geq 1$. It is easily seen that V_2 must contain a point of S, or else $\mathcal{G}$ would not be connected. Then p_L, p_R, and any point in $V_2 \cap S$ form a 1-separated triple. Clearly, finding these points takes linear time. ∎

By Lemmas 3.6 and 3.7, it remains to consider the case in which $\mathcal{G}$ is connected and $\mathcal{C}$ spans at most $3\mu+1$ rows and at most $3\mu+1$ columns. Clearly, in this case $|\mathcal{C}| \leq (3\mu+1)^2$. We consider all triples $C_1, C_2, C_3 \in \mathcal{C}$ and determine whether $S_1 \times S_2 \times S_3$ contains a 1-separated triple, where $S_i = C_i \cap S$, for $i = 1, 2, 3$. If the maximum distance between two of these three cells, say, C_1 and C_2, is less than 1, then no 1-separated triple in $S_1 \times S_2 \times S_3$ exists. Hence, we can assume that the maximum distance between every pair of C_1, C_2, C_3 is at least 1. There are four cases to consider, depending on the number k of edges of $\mathcal{G}$ between these three cells:

(i) $k = 0$; that is, C_1, C_2, C_3 is an independent set in $\mathcal{G}$. Then any triple in $S_1 \times S_2 \times S_3$ is 1-separated, and we return one of these triples.

(ii) $k = 1$; suppose, without loss of generality, that $(C_1, C_2) \in \mathcal{E}$ and $(C_1, C_3), (C_2, C_3) \notin \mathcal{E}$. We compute a diametral pair (p, q) of $S_1 \cup S_2$. If $|pq| \geq 1$, then we return (p, q, r), where r is any point of S_3. If $|pq| < 1$, no triple in $S_1 \times S_2 \times S_3$ is 1-separated. This step takes $O(n \log n)$ time.

(iii) $k = 2$; suppose, without loss of generality, that $(C_1, C_2), (C_1, C_3) \in \mathcal{E}$ and $(C_2, C_3) \notin \mathcal{E}$. We compute $K(S_2)$ and $K(S_3)$. If a point $p \in S_1$ lies neither in $K(S_2)$ nor in $K(S_3)$, then there exists a pair $(q, r) \in S_2 \times S_3$ so that $p \notin \mathbb{D}(q) \cup \mathbb{D}(r)$ and thus (p, q, r) is 1-separated. If $S_1 \subseteq K(S_2) \cup K(S_3)$, then, arguing as in the proof of Lemma 3.1, no triple in $S_1 \times S_2 \times S_3$ is 1-separated. This step too takes $O(n \log n)$ time.

(iv) $k = 3$; that is, $(C_1, C_2), (C_1, C_3), (C_2, C_3) \in \mathcal{E}$. In other words, for any pair $i \neq j \in \{1, 2, 3\}$ we have

$$\min\{|xy| \mid x \in C_i, y \in C_j\} < 1 \leq \max\{|xy| \mid x \in C_i, y \in C_j\}.$$

By the triangle inequality, this implies that any $x \in C_i$, $y \in C_j$ satisfy $1 - 2\sqrt{2}\varepsilon \leq |xy| \leq 1 + 2\sqrt{2}\varepsilon$. We claim that our choice of ε implies that there exist points $c_1, c_2, c_3 \in \mathbb{R}^2$ so that $|c_i c_j| = 1$ for each pair of distinct points c_i, c_j, and S_i is contained in the disk D_i of radius $\delta \leq 1/6$ centered at c_i, for $i = 1, 2, 3$. To see this, pick any pair of points $c_1 \in C_1$, $c_2 \in C_2$, such that $|c_1 c_2| = 1$. Let $c_3 \in \mathbb{R}^2$ be a point such that $\Delta c_1 c_2 c_3$ is equilateral and c_3 lies on the same side of the line through c_1 and c_2 as C_3 (our choice of ε is easily seen to imply that C_3 does not intersect such a line). By what we have just argued, C_3 is fully contained in the intersection of the two annuli

$$1 - 2\sqrt{2}\varepsilon \leq |c_1 x| \leq 1 + 2\sqrt{2}\varepsilon,$$
$$1 - 2\sqrt{2}\varepsilon \leq |c_2 x| \leq 1 + 2\sqrt{2}\varepsilon.$$

A simple calculation then shows that C_3 is fully contained in the disk of radius 1/6 centered at c_3. In other words, in this case S_1, S_2, and S_3 satisfy property ($\triangle$), and we can therefore use Theorem 3.5 to compute a 1-separated triple in $S_1 \times S_2 \times S_3$, if one exists, or to determine that no such triple exists.

The total running time of the algorithm is dominated by the overall cost of handling case (iv), and is thus, by Theorem 3.5, $O(n^{4/3} \log^2 n)$ since $\mu = O(1)$. We thus obtain the following main result of this section.

THEOREM 3.8. *Let S be a set of n points in $\mathbb{R}^2$. We can compute, in $O(n^{4/3} \log^2 n)$ time, a 1-separated triple in S, if one exists, or determine that no such triple exists.*

Finally, we run a binary search on the $\binom{n}{2}$ pairwise distances in S. The kth smallest pairwise distance δ_k in S, for any $1 \le k \le \binom{n}{2}$, can be computed in time $O(n^{4/3} \log^2 n)$ [18], and by Theorem 3.8, we can determine whether a δ_k-separated triple exists in S within the same time bound. Hence, we obtain the following theorem.

THEOREM 3.9. *Let S be a set of n points in $\mathbb{R}^2$. We can compute, in $O(n^{4/3} \log^3 n)$ time, a maximally separated triple in S.*

4. Computing a maximally separated quadruple.

Our overall approach to this problem is similar to the one in Section 3. We first consider a multicolored version of this problem, in which we are given four sets, S_1, S_2, S_3, and S_4, of points, placed "reasonably far" from each other, and we wish to determine whether there exists a 1-separated quadruple in $S_1 \times S_2 \times S_3 \times S_4$. The easy cases are when some pairs of subsets S_i, S_j are either too far from each other or too near each other. The difficult case is when these sets are arranged in a so-called *diamond* configuration, and we present in Section 4.1 below an algorithm for handling this case. We then present the overall algorithm, which, as in the previous section, runs a binary search through the pairwise distances in S, and, for each fixed distance, reduces the general problem to a constant number of multicolored instances.

4.1. The diamond configuration.

Let S_1, S_2, S_3, and S_4 be sets of n points each in $\mathbb{R}^2$ that satisfy the following property:

($\diamond$) There is a constant $\delta \le 1/8$ so that each S_i, for $i = 1, \ldots, 4$, is contained in a disk D_i of radius δ centered at a point c_i, so that $|c_1c_3|, |c_2c_3|, |c_1c_4|, |c_2c_4| = 1$, and $1 \le |c_1c_2| \le 1 + 2\delta < |c_3c_4|$.

Without loss of generality assume that $c_3 = (0, 0)$, c_4 lies on the x-axis to the right of c_3, and c_1 (resp., c_2) lies below (resp., above) the x-axis (in symmetric positions). The conditions on the c_i's imply that

$$1 \le |c_1c_2| \le 1 + 2\delta < \sqrt{2} \le |c_3c_4| \le \sqrt{3}.$$

See Figure 6.

Note that one helpful property of the diamond configuration is that any pair of points in $S_3 \times S_4$ is 1-separated, so only five of the six pairwise distances in a quadruple in $S_1 \times S_2 \times S_3 \times S_4$ need to be considered.

For a point $p \in S_1$, let $S_p^{(3)} = \{q \in S_3 \mid |pq| \ge 1\}$ and $S_p^{(4)} = \{q \in S_4 \mid |pq| \ge 1\}$. (We ignore for the time being the issue of efficient construction of these sets; this will be addressed later on.) We remove from S_1 any point p for which one of these sets is empty, because such a p cannot be part of a 1-separated quadruple in $S_1 \times S_2 \times S_3 \times S_4$.

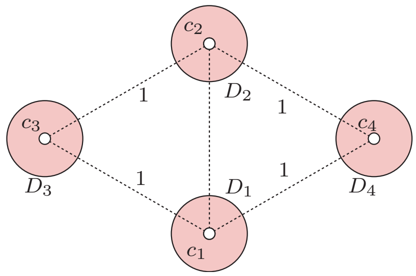

FIG. 6. *A diamond configuration. The missing edge between D_3 and D_4 indicates that every pair of points in $S_3 \times S_4$ is 1-separated.*

Set, for each remaining $p \in S_1$,

$$K_p^{(3)} := \bigcap_{q \in S_p^{(3)}} \mathbb{D}(q), \qquad K_p^{(4)} := \bigcap_{q \in S_p^{(4)}} \mathbb{D}(q),$$
$$R_p := K_p^{(3)} \cup K_p^{(4)} \cup \mathbb{D}(p), \qquad \mathcal{R} := \bigcap_{p \in S_1} R_p.$$

The following lemma is fairly straightforward (cf. Lemma 3.1).

LEMMA 4.1. *There exists a 1-separated quadruple in $S_1 \times S_2 \times S_3 \times S_4$ if and only if $S_2 \not\subseteq \mathcal{R}$.*

Proof. If there exists a point $q \in S_2 \setminus \mathcal{R}$, then there exists a point $p \in S_1$ so that $q \notin R_p = K_p^{(3)} \cup K_p^{(4)} \cup \mathbb{D}(p)$, which implies that there is a pair $(u, v) \in S_p^{(3)} \times S_p^{(4)}$ so that $q \notin \mathbb{D}(u) \cup \mathbb{D}(v) \cup \mathbb{D}(p)$. Therefore, $|pu|, |qu|, |pv|, |qv|, |pq| \geq 1$. By property ($\diamond$), $|uv| \geq 1$ too. Hence, (p, q, u, v) is 1-separated. The converse implication is argued in the same manner. ∎

The following is a variant of Lemma 3.3, proved in a similar manner.

LEMMA 4.2.

(i) *For any $p \in S_1$, $\partial K_p^{(3)}$ and $\partial K_p^{(4)}$ intersect above the x-axis at exactly one point σ_p, which lies on the upper boundaries of both regions.*

(ii) *For any $p \in S_1$, $\partial K_p^{(3)}$ and $\partial \mathbb{D}(p)$ intersect above the x-axis exactly once, and similarly for $\partial K_p^{(4)}$ and $\partial \mathbb{D}(p)$.*

Proof. (i) Let W_3 (resp., W_4) denote the annulus bounded by the concentric circles of radii $1 + \delta$ and $1 - \delta$ centered at c_3 (resp., c_4). By Lemma 3.2, $\partial K_p^{(3)}$ (resp., $\partial K_p^{(4)}$) is contained in W_3 (resp., W_4). Therefore $\partial K_p^{(3)} \cap \partial K_p^{(4)} \subseteq W_3 \cap W_4$. Since $\delta < 1 - \sqrt{3}/2$ and $|c_3 c_4| \leq \sqrt{3}$, the inner circles of W_3 and W_4 intersect and thus $W_3 \cap W_4$ consists of two connected components Σ^+, Σ^-, where Σ^+ lies above the x-axis and Σ^- below the x-axis (as in Figure 3(i)). Moreover, by the choice of δ, Σ^+ lies fully to the right of D_3, to the left of D_4, and above both these disks, as is easily verified. This implies that, within Σ^+, the boundary of each $\mathbb{D}(q)$, for $q \in S_p^{(3)}$, is the graph of a decreasing function, and thus $\partial K_p^{(3)}$ is also the graph of a decreasing function within Σ^+. By a fully symmetric argument, $\partial K_p^{(4)}$ is the graph of an increasing function within Σ^+. Moreover, $\partial K_p^{(3)} \cap \Sigma^+$ is contained in the upper boundary of $K_p^{(3)}$, and similarly for $K_p^{(4)}$, because Σ^+ lies above D_3 and D_4. This is easily seen to imply the assertion in (i).

(ii) The center p of $\mathbb{D}(p)$ lies to the right of the center of the circle of any arc that appears on $\partial K_p^{(3)}$, and only the upper semicircle of $\mathbb{D}(p)$ can be above the x-axis.

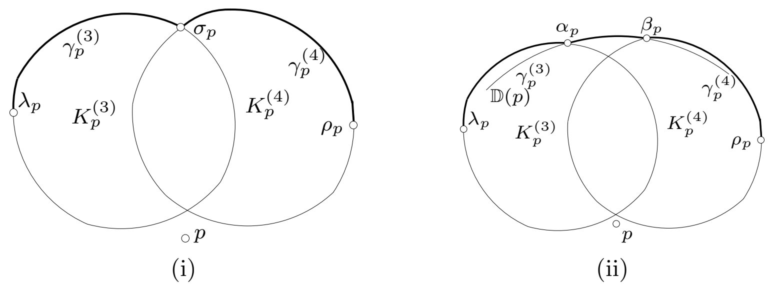

FIG. 7. (i) *Implicit representation of the upper boundary of* R_p *when* $\mathbb{D}(p)$ *does not appear on this boundary.* (ii) *Implicit representation of the upper boundary of* R_p *when* $\mathbb{D}(p)$ *appears on it. In both cases, the boundary is drawn as a thick curve.*

Hence, above the x-axis, $\partial\mathbb{D}(p)$ intersects any arc γ that bounds $\partial K_p^{(3)}$ in at most one point, and lies above γ to the right of that point. Hence $\partial\mathbb{D}(p)$ lies above $\partial K_p^{(3)}$ to the right of any intersection point between the two curves, which readily implies (ii). $\square$

DEFINITION 4.3. *For a point* $p \in S_1$*, let* σ_p *denote, as in Lemma* 4.2*, the unique intersection point of the upper boundaries of* $K_p^{(3)}$ *and* $K_p^{(4)}$*, and let* α_p *(resp.,* β_p*) denote the intersection point of the upper boundary of* $K_p^{(3)}$ *(resp.,* $K_p^{(4)}$*) with* $\mathbb{D}(p)$ *if such a point exists.*

Consider the upper envelope of the upper boundaries of $K_p^{(3)}$, $K_p^{(4)}$, and $\mathbb{D}(p)$. The preceding analysis implies that the envelope has one of the following two structures: (a) Either $\mathbb{D}(p)$ does not appear on the envelope, and then the envelope consists of a connected portion $\gamma_p^{(3)}$ of the upper boundary of $K_p^{(3)}$ and a connected portion $\gamma_p^{(4)}$ of the upper boundary of $K_p^{(4)}$, meeting at the point σ_p (see Figure 7 (i)); or (b) $\mathbb{D}(p)$ appears on the envelope, and then the envelope consists of a connected portion $\gamma_p^{(3)}$ of the upper boundary of $K_p^{(3)}$, a connected portion δ_p of the upper boundary of $\mathbb{D}(p)$, and a connected portion $\gamma_p^{(4)}$ of the upper boundary of $K_p^{(4)}$, so that the first and second portions meet at α_p and the second and third portions meet at β_p (see Figure 7 (ii)).

Let $\Gamma^{(3)} = \{\gamma_p^{(3)} \mid p \in S_1\}$, $\Gamma^{(4)} = \{\gamma_p^{(4)} \mid p \in S_1\}$, and $\Delta = \{\delta_p \mid p \in S_1\}$. Note the difference between this notation and the one in section 3. There Γ_i was a family of circular arcs, whereas here each arc in $\Gamma^{(3)}$ or $\Gamma^{(4)}$ is a sequence of circular arcs. Let $\mathcal{L}^{(3)}$ (resp., $\mathcal{L}^{(4)}$, $\mathcal{L}^{(1)}$) denote the lower envelope of $\Gamma^{(3)}$ (resp., $\Gamma^{(4)}$, Δ).

The following corollary follows from Lemmas 3.4 and 4.2. Its proof uses the obvious observation that the lower envelope of $\mathcal{L}^{(3)}$, $\mathcal{L}^{(4)}$, and $\mathcal{L}^{(1)}$ is the same as the lower envelope of the upper boundaries of the regions R_p, for $p \in S_1$. (We follow here the convention that if a curve is undefined at some x, it is assumed to be $+\infty$ there.) Arguing in much the same way as in Section 3 and exploiting the fact that $\delta < 1/8$, one can show that a point of S_2 does not lie below the lower boundary of any $K_p^{(3)}, K_p^{(4)}$, or $\mathbb{D}(p)$. Hence, we obtain the following corollary.

COROLLARY 4.4. *A point* $q \in S_2$ *lies in* $\mathcal{R}$ *if and only if* q *lies below each of* $\mathcal{L}^{(3)}$*,* $\mathcal{L}^{(4)}$*, and* $\mathcal{L}^{(1)}$*.*

We thus compute each of the envelopes $\mathcal{L}^{(3)}$, $\mathcal{L}^{(4)}$, and $\mathcal{L}^{(1)}$ separately, and determine whether any point of S_2 lies above any of them. If the answer is yes, we can conclude that a 1-separated quadruple in $S_1 \times S_2 \times S_3 \times S_4$ exists, and we can compute it in additional linear time. Otherwise no such quadruple exists.

However, unlike the situation in Section 3, computing these envelopes explicitly is expensive, because they consist of too many arcs, so we represent them implicitly. We first describe the implicit representation of the envelopes and of their arcs, following a similar representation used by Agarwal, Sharir, and Welzl [4], and then present the algorithm for computing and searching in the envelopes.

Implicit representation of $K_p^{(3)}, K_p^{(4)}$*, and of the lower envelopes.* For a subset $Q \subseteq S_1$, let $\mathcal{L}_Q^{(3)}$ denote the lower envelope of arcs in the set $\{\gamma_p^{(3)} \mid p \in Q\}$. We represent $\mathcal{L}_Q^{(3)}$ by the sequence of its *breakpoints* in increasing order of their x-coordinates. The breakpoints are defined so that each portion ξ of $\mathcal{L}_Q^{(3)}$ between two consecutive breakpoints is contained in a single $\gamma_p^{(3)}$ (such a ξ may overlap with many $\gamma_p^{(3)}$'s, but there is (at least) one point $p \in S_1$ such that ξ is fully contained in $\gamma_p^{(3)}$). We maintain ξ implicitly, by recording a point $p \in Q$ that satisfies $\xi \subseteq \gamma_p^{(3)}$. Recall that each $\gamma_p^{(3)}$ may consist of many circular arcs, bounding different disks centered at points of $S_p^{(3)}$; our implicit representation avoids the costly explicit enumeration of these arcs.

Let $\mathcal{D}_3 = \{\mathbb{D}(q) \mid q \in S_3\}$. To represent each $\gamma_p^{(3)}$ implicitly, we choose a parameter $n \le s \le n^2$, and use the result of Katz and Sharir [18], which shows that there exists a family $\mathcal{F}^{(3)} = \{\mathcal{D}^{(1)}, \dots, \mathcal{D}^{(u)}\}$ of *canonical* subsets of $\mathcal{D}$, with a corresponding family of intersection regions $\mathcal{I}^{(i)} = \bigcap \mathcal{D}^{(i)}$, for $i = 1, \dots, u$, such that $\sum_{i=1}^{u} |\mathcal{D}^{(i)}| = O(s \log n)$, and such that for any $p \in S_1$, $K_p^{(3)}$ can be represented as the intersection of $O((n/\sqrt{s}) \log n)$ of these canonical regions $\mathcal{I}^{(i)}$. Let $J_p^{(3)}$ denote the set of indices of these canonical regions, i.e., $K_p^{(3)} = \bigcap_{j \in J_p^{(3)}} \mathcal{I}^{(j)}$. We represent each arc $\gamma_p^{(3)}$ by its endpoints and by the set $J_p^{(3)}$. We also store the vertices of all $\mathcal{I}^{(j)}$ in a single master list Λ, sorted in increasing order of their x-coordinates.

As shown in [18], $\mathcal{F}^{(3)}$ and the sets $J_p^{(3)}$ for all $p \in S_3$ can be computed in time $O((s + n^2/\sqrt{s}) \log n)$, and we spend another $O(s \log^2 n)$ time to compute the regions $\mathcal{I}^{(j)}$. A similar representation was developed by Agarwal, Sharir, and Welzl [4], who showed that the above representation enables us to perform each of the following four operations on $\{\partial K_p^{(3)}\}_{p \in S_1}$ in $O((n/\sqrt{s}) \log^3 n)$ time.

(S1) *Leftmost and rightmost points:* Given a point $p \in S_1$, compute the leftmost and rightmost points of $K_p^{(3)}$.

(S2) *Intersection point(s) with a vertical line:* Given a vertical line ℓ and a point $p \in S_1$, determine the intersection point(s) of ℓ with $\partial K_p^{(3)}$.

(S3) *Intersection points with a unit disk:* Given a unit disk $\mathbb{D}$ and a point $p \in S_1$, determine the intersection point(s) of $\mathbb{D}$ with $\partial K_p^{(3)}$.

(S4) *Crossing point of two arcs:* Given two points $p, q \in S_1$ and an x-interval $[a, b]$ contained in the x-span of the top boundaries of $K_p^{(3)}$ and $K_q^{(3)}$, determine whether they cross in $[a, b]$. If so, return their crossing point. If they *weakly cross* in $[a, b]$, i.e., overlap over some subinterval J of $[a, b]$ and their vertical order to the right of J is the reverse of their vertical order to the left of J, then return the leftmost endpoint of their common overlap in $[a, b]$.

In a fully analogous fashion, we process S_4 in $O((s + (n^2/\sqrt{s})) \log n)$ time, to compute an implicit representation of all the arcs $\gamma_p^{(4)}$. Each of the operations (S1)–(S4) on $\{\partial K_p^{(4)}\}_{p \in S_1}$ can also be performed in $O((n/\sqrt{s}) \log^3 n)$ time. Moreover, given any $p \in S_1$, the intersection point of the top boundaries of $K_p^{(3)}$ and $K_p^{(4)}$ can also be computed in $O((n/\sqrt{s})\log^3 n)$ time.

Computing $\mathcal{L}^{(3)}$, $\mathcal{L}^{(4)}$, *and* $\mathcal{L}^{(1)}$. Using the subroutine (S1), we first compute the lefmost point λ_p of $K_p^{(3)}$ and the rightmost point ρ_p of $K_p^{(4)}$, for each $p \in S_1$. Next, using (S3) and (S4), we compute the intersection point σ_p of the upper boundaries of $K_p^{(3)}$ and $K_p^{(4)}$, the intersection point α_p of the upper boundaries of $K_p^{(3)}$ and $\mathbb{D}(p)$, and the intersection point β_p of the upper boundaries of $K_p^{(4)}$ and $\mathbb{D}(p)$. By comparing the x-coordinates of these three points, we can determine whether the upper boundary of R_p is of the first kind (disjoint from $\mathbb{D}(p)$) or of the second kind (overlapping an arc of $\mathbb{D}(p)$). In the first case, $\gamma_p^{(3)}$ (resp., $\gamma_p^{(4)}$) is the portion of the upper boundary of $K_p^{(3)}$ (resp., $K_p^{(4)}$) between λ_p and σ_p (resp., σ_p and ρ_p). In the second case it is the portion of $\partial K_p^{(3)}$ (resp., $\partial K_p^{(4)}$) between λ_p and α_p (resp., β_p and ρ_p). We thus have the endpoints of γ_p and its implicit representation at our disposal.

The x-coordinate of any point $p \in S_1$ is at most $\sqrt{3}/2 + \delta$ and the x-coordinate of the rightmost point of R_p is at least $1 - \delta$. This easily implies that p lies below $K_p^{(3)}$ (i.e., the vertical ray emanating upward from p intersects $K_p^{(3)}$). Since this holds for any point $p \in S_1$, Theorem 2.8 of Agarwal, Sharir, and Welzl [4] (concerning the "pseudo-segment" property of the upper portions of $\partial K_p^{(3)}$) implies that, for any $p, q \in S_1$, $\gamma_p^{(3)}$ and $\gamma_q^{(3)}$ cross in at most one point. A similar argument proves the corresponding claim for the curves in $\Gamma^{(4)}$. Hence, each of $\Gamma^{(3)}, \Gamma^{(4)}$ is a collection of pseudo-segments. We can therefore compute the lower envelopes $\mathcal{L}^{(3)}, \mathcal{L}^{(4)}$ using the divide-and-conquer algorithm of Hershberger [16], mentioned above. In the main step of this algorithm, we have envelopes $\mathcal{L}_A^{(3)}, \mathcal{L}_B^{(3)}$ of two subsets $A, B \subseteq \Gamma^{(3)}$ at our disposal, and we need to merge these envelopes to compute $\mathcal{L}_{A\cup B}^{(3)}$. The only nontrivial part in the merge step is computing the crossing point of two arcs $\gamma_p^{(3)}$ and $\gamma_q^{(3)}$ in a given x-interval $[a, b]$ that is contained in the x-span of both $\gamma_p^{(3)}$ and $\gamma_q^{(3)}$. Using the subroutine (S4), we can compute, in $O((n/\sqrt{s})\log^3 n)$ time, the crossing point of the upper boundaries of $K_p^{(3)}$ and $K_p^{(4)}$ in the interval $[a, b]$, if it exists. If so, this is the crossing point of $\gamma_p^{(3)}$ and $\gamma_q^{(3)}$; otherwise, these arcs do not intersect over $[a, b]$. Plugging this bound into Hershberger's algorithm, we can compute an implicit representation of $\mathcal{L}^{(3)}$ in overall time $O((n^2/\sqrt{s})\log^4 n + s\log^2 n)$. Similarly, we can compute an implicit representation of $\mathcal{L}^{(4)}$ within the same time bound $O((n^2/\sqrt{s})\log^4 n + s\log^2 n)$. Computing $\mathcal{L}^{(1)}$ is easier, since no implicit representation is needed here: We simply have to compute the lower envelope of at most n upper unit circular arcs that behave as pseudo-segments, so their envelope can be computed in $O(n\log n)$ time (as in [16]). Finally, for each point $q \in S_2$, we determine in $O((n/\sqrt{s})\log^3 n)$ time, using the subroutine (S2), whether q lies above $\mathcal{L}^{(3)}$ or above $\mathcal{L}^{(4)}$. Testing whether q lies above $\mathcal{L}^{(1)}$ is easy to accomplish in $O(\log n)$ time. The total time spent is thus $O\left((n^2/\sqrt{s})\log^4 n + s\log^2 n\right)$. By choosing $s = n^{4/3}\log^{4/3} n$, we obtain the following summary result.

THEOREM 4.5. *Let S_1, S_2, S_3, and S_4 be four sets of n points each in $\mathbb{R}^2$ that satisfy property ($\diamond$). One can determine, in time $O(n^{4/3}\log^{10/3} n)$, whether $S_1 \times S_2 \times S_3 \times S_4$ contains a 1-separated quadruple, and, if so, compute such a quadruple.*

4.2. Reduction to the multicolored case. As in section 3, we construct a square grid of size ε, for a sufficiently small constant parameter $\varepsilon > 0$. Let $C_{ij}, S_{ij}, \mathcal{C}, \mathcal{G}$, and μ be as in section 3.

LEMMA 4.6. *If $\mathcal{G}$ is not connected, then we can compute a* 1*-separated quadruple in S (or determine that no such quadruple exists) in $O(n^{4/3}\log^2 n)$ time.*

Proof. If $\mathcal{G}$ has at least two connected components, then let $S_1 \subseteq S$ be the subset of points lying in the grid cells of one connected component, and put $S_2 := S \setminus S_1$. If a 1-separated quadruple exists, then there also exists a 1-separated quadruple that has points in both S_1 and S_2. Indeed, if (p_1, p_2, p_3, p_4) is a 1-separated quadruple that is contained in, say, S_1, then the quadruple obtained by replacing, say, p_1 by any point of S_2 is also 1-separated. Hence it suffices to look for 1-separated quadruples that have two points in each of S_1, S_2, or have three points in one of these sets and one point in the other set. Moreover, it suffices to find the two parts of such a quadruple *independently*—putting together any pair of such parts, one contained in S_1, the other in S_2, and consisting together of four points, will form a 1-separated quadruple in S. In the former case, it suffices to check that $\min\{\mathrm{diam}(S_1), \mathrm{diam}(S_2)\} \geq 1$, and then return a pair of diametral points in each of S_1, S_2. In the latter case, we apply the decision procedure of Section 3 to S_1 and to S_2. If either of these applications yields a 1-separated triple, combining it with any point in the other set yields a 1-separated quadruple in S. If none of these steps succeeds, S has no 1-separated quadruple. The overall cost of the procedure just sketched is, by Theorem 3.8, $O(n^{4/3}\log^2 n)$. ◻

LEMMA 4.7. *If $\mathcal{G}$ is connected and $\mathcal{C}$ spans more than $5\mu+1$ columns or rows of the grid, i.e., it has cells in two columns (or rows) whose indices j, j' satisfy $j'-j \geq 5\mu+1$, then a* 1*-separated quadruple in S exists, and can be constructed in $O(n)$ time.*

Proof. Consider the case where $\mathcal{C}$ spans more than $5\mu + 1$ columns. Let $C_{i_L j_L}$ (resp., $C_{i_R j_R}$) be a grid cell of $\mathcal{C}$ in the leftmost (resp., rightmost) column, and let $p_L \in S_{i_L j_L}$, and $p_R \in S_{i_R j_R}$. By assumption, $j_R - j_L \geq 5\mu + 1$. We group the columns between the j_Lth and j_Rth columns (exclusive) into five pairwise-disjoint vertical strips $V_1, \ldots, V_5$, appearing in this left-to-right order, each of width at least $\mu\varepsilon \geq 1$. We argue that each of V_2, V_4 must contain points of S, or else $\mathcal{G}$ would not be connected. Then p_L, p_R, any point in $V_2 \cap S$, and any point in $V_4 \cap S$ form a 1-separated quadruple. Clearly, finding these points takes linear time. ◻

By Lemmas 4.6 and 4.7, we may therefore assume that $\mathcal{G}$ is connected and that $\mathcal{C}$ spans at most $5\mu + 1$ rows and at most $5\mu + 1$ columns. In this case, we try all quadruples $C_1, C_2, C_3, C_4 \in \mathcal{C}$ and determine whether the corresponding product $S_1 \times S_2 \times S_3 \times S_4$ contains a 1-separated quadruple. We can assume that, for each pair of cells, the maximum distance between points in these two cells is at least one, and that the subgraph induced by these four cells is connected, because if the former assumption is violated, then no 1-separated quadruple exists, and if the latter is violated, then we can find a 1-separated quadruple (or determine that none exists), proceeding as in Lemma 4.6. In other words, we may assume that, for each C_i, C_j,

$$1 \leq \max_{x \in C_i,\, y \in C_j} |xy| \leq 1 + 2\sqrt{2}\varepsilon.$$

We now proceed by case analysis, according to the structure of the edges of $\mathcal{G}$ that connect the cells $C_1, \ldots, C_4$. Figure 8 shows all the possible cases, up to symmetries. It is easily checked that the complete graph on $C_1, \ldots, C_4$ is impossible, if ε is chosen sufficiently small.

In cases (i), (ii), and (iii), there is at least one node that has degree 1 in $\mathcal{G}$. It is then easy to reduce the problem to the case of finding a 1-separated triple. Consider for example case (iii), where the only edge incident to C_2 is (C_2, C_3). We then replace

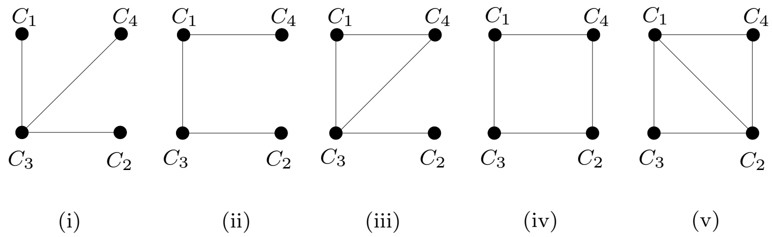

FIG. 8. *Possible graphs for $k = 4$.*

S_3 by the set

$$S_3' = \Big\{p \in S_3 \mid \max_{q \in S_2} |pq| \geq 1\Big\}.$$

S_3' can be computed in time $O(n \log n)$, by constructing $K(S_2)$ and choosing all points of S_3 that lie outside $K(S_2)$. We now find, in time $O(n^{4/3} \log^2 n)$, a 1-separated triple in $S_1 \times S_2 \times S_3'$, or determine that none exists. Once such a triple (p, q, r) is found, any point $s \in S_2$ for which $|rs| \geq 1$ can be added to it to form a 1-separated quadruple; s can be found in additional $O(n)$ time.

This leaves us with cases (iv) and (v). Pick points $c_i \in C_i$, for $i = 1, 2, 3, 4$, such that $|c_1c_3| = |c_2c_4| = 1$. Then

$$1 - 2\sqrt{2}\varepsilon \leq |c_2c_3|, |c_1c_4| \leq 1 + 2\sqrt{2}\varepsilon.$$

Hence, by translating c_2c_4 by distance $\leq 2\sqrt{2}\varepsilon$, we can also enforce $|c_2c_3| = 1$, and $1-4\sqrt{2}\varepsilon \leq |c_1c_4| \leq 1+4\sqrt{2}\varepsilon$. Note that $\angle c_1c_3c_2$ and $\angle c_3c_2c_4$ cannot be much smaller than $\pi/3$ each, because otherwise c_1 and c_2, or c_3 and c_4, would be too close to each other, contrary to what we are assuming. For the same reason, these angles cannot be much larger than $2\pi/3$.

This is easily seen to imply that, by slightly rotating c_4 around c_2, we can make the distance $|c_1c_4|$ also equal to 1. The new points c_i are no longer necessarily inside the respective cells C_i, for $i = 2, 3, 4$, but they remain close to these cells. If ε is chosen sufficiently small, the disk of radius $\delta = 1/8$ around c_i will fully contain C_i, for $i = 1, \ldots, 4$. Moreover, by slightly flexing the rhombus $c_1c_3c_2c_4$, we can also assume that $|c_1c_2| \geq 1$, while the containment property just mentioned continues to hold. If $|c_1c_2| \leq 1 + 2\delta = 5/4$, then S_1, S_2, S_3, S_4 satisfy property ($\diamond$). In this case, we can apply the algorithm of Theorem 4.5 to find a 1-separated quadruple in $S_1 \times S_2 \times S_3 \times S_4$, or to determine that none exists, in time $O(n^{4/3} \log^{10/3} n)$.

If $|c_1c_2| > 5/4$, any pair of points $p \in S_1$, $q \in S_2$ is 1-separated. We can then apply a simpler variant of the algorithm in section 4.1, in which we ignore any interaction between S_1 and S_2. Thus we may ignore the family Δ of disks, and only consider the intersection points σ_p and not α_p, β_p. Alternatively, we can run the algorithm as is, and the disks $\mathbb{D}(p)$, for $p \in S_1$, will never show up on the overall envelope. In either case, the running time is $O(n^{4/3} \log^{10/3} n)$.

In summary, we show the following theorem.

THEOREM 4.8. *Let S be a set of n points in $\mathbb{R}^2$. A 1-separated quadruple in S can be computed (or be determined not to exist) in time $O(n^{4/3} \log^{10/3} n)$.*

Finally, by performing a binary search on the pairwise distances in S, as in section 3, we obtain the following main result of this section.

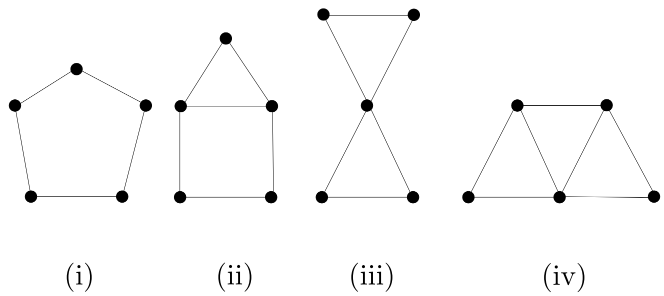

FIG. 9. *Possible graphs for* $k = 5$.

THEOREM 4.9. *Let S be a set of n points in $\mathbb{R}^2$. A maximally separated quadruple in S can be computed in $O(n^{4/3} \log^{13/3} n)$ time.*

4.3. Discussion. The technique that we have presented in sections 3 and 4 can be extended in principle to larger values of k. As above, it suffices to solve the decision problem: Determine whether a 1-separated k-tuple exists in S. Lemmas 4.6 and 4.7 can be extended in a straightforward manner, and they reduce the problem to $O(1)$ subproblems. In each subproblem we have k $\varepsilon \times \varepsilon$ square cells $C_1, \ldots, C_k$ of a grid, and subsets $S_i = S \cap C_i$, for $i = 1, \ldots, k$. Every pair of cells is such that the maximum distance between their points is at least 1, and some pairs of cells are such that the minimum distance between their points is at most 1. The collection of the pairs of the second kind constitutes the edge set of a graph $\mathcal{G}$, and the problem proceeds by case analysis, depending on the structure of $\mathcal{G}$. As above, we may assume that $\mathcal{G}$ is connected, and that the degree of each node is at least two.

For example, consider the case $k = 5$. The possible graphs $\mathcal{G}$ that need to be considered are shown in Figure 9. We leave it as an open problem to design efficient algorithms for the decision problem on each of these graphs, and thus to obtain an efficient algorithm for finding a maximally separated 5-tuple in S.

5. An exact algorithm for an arbitrary *k*. Let S be a set of n points in $\mathbb{R}^2$, and let $k \geq 2$ be an integer. We describe an $n^{O(\sqrt{k})}$-time algorithm for computing a maximally separated subset of S of size k. As in the previous sections, it suffices to focus on the decision problem: Given a set $\mathcal{D}$ of n unit disks and an integer $1 \leq k \leq n$, is there a subset $I \subseteq \mathcal{D}$ of k pairwise-disjoint disks?

Suppose that all the disks of $\mathcal{D}$ lie inside a horizontal strip W of (integer) width w. Using a sweep-line algorithm, similar to the one by Gonzalez [14] for computing a k-center of a set of points, we can compute a largest subset of pairwise-disjoint disks in $n^{O(w)}$ time, as follows.

We define the *index* of a set $A = \{D_1, \ldots, D_q\}$ of unit disks, for $q \leq n$, to be the $2n$-vector

$$\sigma(A) = (\underbrace{0, \ldots, 0, x_1, \ldots, x_q}_{n}, \underbrace{0, \ldots, 0, y_1, \ldots, y_q}_{n}),$$

where (x_i, y_i) is the center of D_i, $x_1 \leq x_2 \leq \cdots \leq x_q$, and if $x_i = x_{i+1}$, then $y_i < y_{i+1}$. We refer to the set of pairwise-disjoint disks with the maximal index in lexicographic order as the *optimal independent set*. The sweep-line algorithm computes the optimal independent set of $\mathcal{D}$, as follows.

For a subset $A \subseteq \mathcal{D}$ and a vertical line ℓ, let $\chi(A, \ell) \subseteq A$ be the set of disks in A that intersect ℓ. We sweep a vertical line ℓ from left to right, stopping at the leftmost and rightmost point of each disk in $\mathcal{D}$. At any time, the algorithm maintains a family $\mathcal{F} = \{I_1, \ldots, I_u\}$ of subsets of pairwise-disjoint disks that satisfies the following invariants:

(I.1) For every $1 \le j \le u$, no disk in $I_j \subseteq \mathcal{D}$ is contained in the (closed) halfplane lying to the right of the sweep line.

(I.2) For $a \ne b$, $\chi(I_a, \ell) \ne \chi(I_b, \ell)$.

(I.3) If there is a subset $A \subseteq \mathcal{D}$ of pairwise-disjoint disks so that no disk in A lies completely to the right of ℓ, then there is a subset $I_j \in \mathcal{F}$ so that $\chi(I_j, \ell) = \chi(A, \ell)$ ($\chi(I_j, \ell)$ may be empty) and $\sigma(A) \le_{\text{lex}} \sigma(I_j)$.

Since at most $O(w)$ pairwise-disjoint disks of $\mathcal{D}$ can intersect ℓ, invariant (I.2) implies that $|\mathcal{F}| = n^{O(w)}$ throughout the sweep. When the sweep line ℓ passes through the leftmost point of a disk $D \in \mathcal{D}$, we test, for each set $I \in \mathcal{F}$, whether D does not intersect any disk in I, and, if so, we add the set $I \cup \{D\}$ to $\mathcal{F}$ (and we also keep I in $\mathcal{F}$). When ℓ passes through the rightmost point of a disk D, we delete all the sets I_a from $\mathcal{F}$ for which there is another set $I_b \in \mathcal{F}$ with $\sigma(I_a) <_{\text{lex}} \sigma(I_b)$ and $\chi(I_a, \ell) \setminus \{D\} = \chi(I_b, \ell) \setminus \{D\}$. We spend $O(|\mathcal{F}|^2 n)$ time at each leftmost or rightmost point of a disk, so the overall running time remains $n^{O(w)}$ (with an appropriate calibration of the constant of proportionality in the exponent). The reader can easily verify that the invariants (I.1)–(I.3) ensure that the algorithm computes the optimal independent set of $\mathcal{D}$.

If $w \le 2\sqrt{k}+2$, we use the above algorithm to determine, in $n^{O(\sqrt{k})}$ time, whether $\mathcal{D}$ contains k pairwise-disjoint disks. So assume that $w > 2\sqrt{k} + 2$. Let Π be the set of at most $2n$ horizontal lines tangent to disks in $\mathcal{D}$. The following packing lemma was proved by Agarwal and Procopiuc [3].

LEMMA 5.1. *Let $\mathcal{D}$ be a set of n disks in $\mathbb{R}^2$, and let $\mathcal{D}' \subseteq \mathcal{D}$ be a subset of at most k pairwise-disjoint unit disks that lie in a horizontal strip of width larger than $2\sqrt{k} + 2$. Then there exists a horizontal line tangent to one of the disks in $\mathcal{D}$ that intersects at most $\sqrt{k}$ disks of $\mathcal{D}'$.*

For any $h \in \Pi$ and $I \subseteq \mathcal{D}$, we call (h, I) a *canonical pair* if $|I| \le \sqrt{k}$, the disks in I are pairwise disjoint, and all disks in I intersect h (in general, h may also intersect other disks of $\mathcal{D}$). We define a *c-strip* to be a triple $\tau = (\omega, A_1, A_2)$, where ω is a strip bounded by two lines $\ell_1, \ell_2 \in \Pi$, with ℓ_1 lying above ℓ_2, and (ℓ_1, A_1) and (ℓ_2, A_2) are canonical pairs; A_1, A_2 are not necessarily disjoint. Let $\mathcal{D}_\tau \subseteq \mathcal{D}$ be the set of disks that do not intersect ℓ_1, ℓ_2 or any disk of $A_1 \cup A_2$. We define the optimal independent set of τ, denoted by $\mathbb{I}_\tau$, to be the optimal independent set of $\mathcal{D}_\tau$, and set $\kappa_\tau := |\mathbb{I}_\tau|$. We call τ *thin* if the width of ω is at most $2\sqrt{k} + 2$, and *thick* otherwise.

For a given $\tau = (\omega, A_1, A_2)$, we compute $\mathbb{I}_\tau$ as follows. If τ is thin, then we compute $\mathbb{I}_\tau$ using the sweep-line algorithm described above. So assume that τ is thick. If $\mathcal{D}_\tau \ne \emptyset$, then by Lemma 5.1, there exists a canonical pair (h, I) so that h divides ω into two strips ω^+, ω^- each of width less than that of ω. Let $\tau^+ = (\omega^+, A_1, I)$ and $\tau^- = (\omega^-, I, A_2)$. We compute $\mathbb{I}_{\tau^+}$ and $\mathbb{I}_{\tau^-}$ recursively, and output $\mathbb{I}_\tau := \mathbb{I}_{\tau^+} \cup I \cup \mathbb{I}_{\tau^-}$. Since we do not know the true canonical pair (h, I), we try all canonical pairs and choose the one for which the solution has the largest index. Moreover, instead of solving the problem recursively, we use a bottom-up approach based on dynamic programming.

In particular, we build a table, each of whose entries corresponds to a c-strip $\tau = (w, A_1, A_2)$ and stores κ_τ and $\sigma_\tau = \sigma(\mathbb{I}_\tau)$. If we ever encounter an entry with

$\kappa_\tau > k$, we can conclude that the size of the largest independent set in $\mathcal{D}$ is greater than k, and we restart the algorithm with a new larger value of k. So we assume that $\kappa_\tau \le k$ for all entries. We fill the entries of the table as follows. If $\mathcal{D}_\tau = \emptyset$, we set $\mathbb{I}_\tau := \emptyset$, and if τ is thin, we compute $\mathbb{I}_\tau$ using the sweep-line algorithm and fill the entry. Otherwise, we compute all canonical pairs (h, I) for which h lies inside ω, and let ω^+ (resp., ω^-) be the portion of ω lying above (resp., below), and let $\tau^+ = (\omega^+, A_1, I)$ and $\tau^- = (\omega^-, I, A_2)$. Compute

$$\kappa_\tau := \max_{(h,I)} \left\{\kappa_{\tau^+} + \kappa_{\tau^-} + |I \setminus (A_1 \cup A_2)|\right\},$$

where the maximum is taken over all canonical pairs. Let (h^*, I^*) be the canonical pair for which the maximum is attained. Then $\sigma(\tau)$ is the index of $\mathbb{I}_{\tau^*_+} \cup \mathbb{I}_{\tau^*_-} \cup I^*$, where τ^*_+, τ^*_- are the c-strips defined by the canonical pair (h^*, I^*). If the maximum value of κ_τ is attained by more than one canonical pair, we choose the one for which $\sigma(\tau)$ is maximal.

Let $(\omega_1, \dots, \omega_M)$, for $M = O(n^2)$, be the sequence of horizontal strips determined by pairs of lines in Π, sorted in nondecreasing order of their widths. We fill out the entries of the table having ω_i as the first component of the index before filling the entries with ω_{i+1} as their first component. If $\mathcal{D}$ contains an independent set of size at most k, then the optimal solution for the c-strip $(\omega_M, \emptyset, \emptyset)$ gives the size and index of the optimal independent set of $\mathcal{D}$. Since there are $n^2 \cdot n^{O(\sqrt{k})} \cdot n^{O(\sqrt{k})}$ c-strips, and each entry can be filled in $n^{O(\sqrt{k})}$ time, the overall running time of the decision algorithm is $n^{O(\sqrt{k})}$ (again, with calibration of the constant of proportionality). We thus obtain the following theorem.

THEOREM 5.2. *Let $\mathcal{D}$ be a set of n unit disks in $\mathbb{R}^2$. The optimal independent set $\mathbb{I}$ of $\mathcal{D}$ can be computed in time $n^{O(\sqrt{k})}$, where k is the size of $\mathbb{I}$.*

Returning to the problem of computing a maximally separated subset of S of size k, we perform a binary search on the pairwise distances in S. At each step, we need to determine whether there exists a maximally separated subset $I \subseteq S$ of size k with $d_{\mathrm{sep}}(I) \ge r$, for a given r, which reduces to determining whether there is an independent set of size k in the set $\{D(p, r) \mid p \in S\}$. Using Theorem 5.2, we thus obtain the following theorem.

THEOREM 5.3. *Let S be a set of n points in $\mathbb{R}^2$, and let $1 \le k \le n$ be an integer. A maximally separated subset of S of size k can be computed in $n^{O(\sqrt{k})}$ time.*

6. Conclusion. In this paper we have presented efficient exact and approximation algorithms for computing maximally separated subsets of a set of points in the plane. The approximation algorithm runs in linear time for fixed values of k, and the exact algorithm runs in time $n^{O(\sqrt{k})}$. We also presented $O(n^{4/3}\mathrm{polylog}(n))$-time algorithms for $k = 3, 4$. As mentioned in Section 4.3, it is not clear whether our approach gives a subquadratic algorithm for $k = 5$. Another interesting open problem is whether a maximally separated subset of size k can be computed in time $n^{O(1)} + k^{O(\sqrt{k})}$.

REFERENCES

[1] P. K. AGARWAL, S. HAR-PELED, AND K. R. VARADARAJAN, *Approximating extent measures of points*, J. ACM, 51 (2004), pp. 606–635.
[2] P. K. AGARWAL, M. VAN KREVELD, AND S. SURI, *Label placement by maximum independent set in rectangles*, Comput. Geom., 11 (1998), pp. 209–218.

[3] P. K. Agarwal and C. M. Procopiuc, *Exact and approximation algorithms for clustering*, Algorithmica, 33 (2002), pp. 201–226.
[4] P. K. Agarwal, M. Sharir, and E. Welzl, *The discrete* 2*-center problem*, Discrete Comput. Geom., 20 (1998), pp. 287–305.
[5] J. Alber and J. Fiala, *Geometric separation and exact solutions for the parameterized independent set problem on disk graphs*, J. Algorithms, 52 (2004), pp. 134–151.
[6] R. Boppana and M. M. Halldórsson, *Approximating maximum independent sets by excluding subgraphs*, BIT, 32 (1992), pp. 180–196.
[7] T. M. Chan, *Polynomial-time approximation schemes for packing and piercing fat objects*, J. Algorithms, 46 (2003), pp. 178–189.
[8] T. M. Chan, *A note on maximum independent sets in rectangle intersection graphs*, Inform. Process. Lett., 89 (2004), pp. 19–23.
[9] B. N. Clark, C. J. Colbourn, and D. S. Johnson, *Unit disk graphs*, Discrete Math., 86 (1990), pp. 165–177.
[10] M. de Berg, M. van Kreveld, M. H. Overmars, and O. Schwarzkopf, Computational Geometry: Algorithms and Applications, 2nd ed., Springer-Verlag, Berlin, 2000.
[11] T. Erlebach, K. Jansen, and E. Seidel, *Polynomial-time approximation schemes for geometric intersection graphs*. SIAM J. Comput., 34 (2005), pp. 1302–1323.
[12] M. Formann and F. Wagner, *A packing problem with applications to lettering of maps*, in Proceedings of the 7th Annual Symposium on Computational Geometry, North Conway, NH, 1991, pp. 281–288.
[13] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman, San Francisco, 1979.
[14] T. Gonzalez, *Clustering to minimize the maximum intercluster distance*, Theoret. Comput. Sci., 38 (1985), pp. 293–306.
[15] J. Hastad, *Clique is hard to approximate within* $n^{1-\varepsilon}$, Acta Math., 182 (1999), pp. 105–142.
[16] J. Hershberger, *Finding the upper envelope of* n *line segments in* $O(n \log n)$ *time*, Inform. Process. Lett., 33 (1989), pp. 169–174.
[17] H. B. Hunt III, M. V. Marathe, V. Radhakrishnan, S. S. Ravi, D. J. Rosenkrantz, and R. E. Stearns, *NC-approximation schemes for NP- and PSPACE-hard problems for geometric graphs*, J. Algorithms, 26 (1998), pp. 238–274.
[18] M. Katz and M. Sharir, *An expander-based approach to geometric optimization*, SIAM J. Comput., 26 (1997), pp. 1384–1408.
[19] M. V. Marathe, H. Beru, H. B. Hunt III, S. S. Ravi, and D. J. Rosenkrantz, *Simple heuristics for unit graphs*, Networks, 25 (1995), pp. 59–68.
[20] D. Marx, *Efficient approximation schemes for geometric problems?*, in Proceedings of the 13th Annual European Symposium Algorithm, Palma de Mallorca, Spain, 2005, pp. 448–459.
[21] M. Sharir and P. K. Agarwal, *Davenport-Schinzel Sequences and Their Geometric Applications*, Cambridge University Press, Cambridge, UK, 1995.
[22] E. J. van Leeuwen, *Approximation algorithms for unit disk graphs*, Proceedings of the 31st International Workshop on Graph Theoretic Algorithms, Metz, France, 2005, pp. 351–361.
[23] J. Vleugels and R. C. Veltkamp, *Efficient image retrieval through vantage objects*, Pattern Recognition, 35 (2002), pp. 69–80.