Discrete Comput Geom 5:449–483 (1990)

Discrete & Computational Geometry
© 1990 Springer-Verlag New York Inc.

Partitioning Arrangements of Lines I: An Efficient Deterministic Algorithm*

Pankaj K. Agarwal[1]

Courant Institute of Mathematical Sciences,
New York University, NY 10012, USA

Abstract. In this paper we consider the following problem: Given a set $\mathscr{L}$ of n lines in the plane, partition the plane into $O(r^2)$ triangles so that no triangle meets more than $O(n/r)$ lines of $\mathscr{L}$. We present a deterministic algorithm for this problem with $O(nr \log n \log^{\omega} r)$ running time, where ω is a constant < 3.33.

1. Introduction

In the last few years several randomized divide-and-conquer algorithms for a variety of geometric problems have been developed [AS], [Cl1], [Cl2], [Cl3], [CS], [CTV], [EGS], [GOS] using ε-nets [HW] or the random-sampling technique of [Cl2] (see also [RS1] and [RS2]). The ε-net theory shows that, for a given set X of n objects and a set $\mathscr{R} \subseteq 2^X$ of *ranges* with finite *Vapnik–Chervonenkis dimension* (see [HW] for definition), there exists a subset $N \subset X$ of size r, for any $r > 0$, such that if $\tau \in \mathscr{R}$ and $\tau \cap N = \varnothing$, then $|\tau| = O((n/r) \log r)$. Moreover, the theory asserts that a random sample N of size r will be an ε-net with high probability. A similar result has been independently obtained by Clarkson [Cl2], [Cl3]. It states roughly that if we draw a random sample N of size r from our set X and partition the underlying geometric space into cells with the property that (i) each cell can be defined in terms of only a "constant" number of elements of N, and (ii) each cell does not meet any element of N, then, with high probability, no

* Work on this paper has been supported by Office of Naval Research Grant N00014-87-K-0129, by National Science Foundation Grant DCR-83-20085, and by grants from the Digital Equipment Corporation and the IBM Corporation. A preliminary version of this paper appears in the *Proceedings of the 5th Annual Symposium on Computational Geometry*, 1989, pp. 11–22.

[1] Current address: Department of Computer Science, Duke University, Durham, NC 27706, USA.

cell meets more than $O((n/r)\log r)$ elements of X. Random-sampling techniques allow us to split the original problem into subproblems of small size, which can then be solved recursively. For example, if the objects are "lines in the plane" and an element of $\mathscr{R}$ is the "set of lines intersecting a given triangle," then these results imply that, for a given set $\mathscr{L}$ of n lines in the plane, there exists a subset $N \subset \mathscr{L}$ of size r such that any triangle which misses all lines in N intersects $O((n/r)\log r)$ lines of $\mathscr{L}$. A problem involving the set $\mathscr{L}$ of lines can then be split into subproblems by computing the arrangement of the lines in N and by triangulating every face of the arrangement; each resulting triangle Δ induces a subproblem involving only $O((n/r)\log r)$ lines of $\mathscr{L}$ intersecting Δ. As an example of a problem that benefits from this set-up, consider *Hopcroft's problem* (see [CSY]): Given m points and n lines in the plane, does any point lie on any line? Applying the above divide-and-conquer strategy to the given lines, and partitioning the points among the triangles, each subproblem involves only the points that fall within some triangle and the line cutting it. More details concerning this and other problems of this kind are given in the second part of this paper [A]. The same approach works for other geometric objects in the plane as well as in higher dimensions.

Unfortunately the random-sampling technique or the ε-net theory only proves the existence of such a subset N, but does not show how to construct it deterministically (and efficiently). This has forced all algorithms based on these techniques to use randomization to obtain N (the only randomized step in most of these algorithms is that of choosing a good sample N). The randomization is justified because the above theories show that most subsets N of size r satisfy the desired properties, so a random choice of such a subset will succeed with high probability. Still, a major open problem in this area is to find an efficient deterministic algorithm to obtain a "good" sample. Recently, Chazelle and Friedman [CF] gave a general framework which unifies the results of Haussler and Welzl [HW] and Clarkson [Cl3], and yields a deterministic algorithm to construct a good sample. Although their algorithm runs in polynomial time, it is not very efficient (in particular its time complexity is too high an overhead for most of the applications). This has motivated researchers to look for special cases, where faster deterministic constructions are possible. Woeginger [Wo] gave an $O(n)$ algorithm to construct an ε-net for a very special case, where the objects are points in the plane and the ranges are half-planes. A much more significant result in this direction has been obtained by Matoušek [Ma], who showed that, given a set of n lines in the plane, the plane can be partitioned into $O(r^2)$ triangles in time $O(nr^2\log^2 r)$ so that no triangle intersects more than $O(n/r)$ lines (however, these triangles in general are not formed by an arrangement of a sample subset of the lines). Note that an important property of the algorithm is that each triangle it produces is cut by only $O(n/r)$ lines, rather than $O((n/r)\log r)$, as promised by the probabilistic techniques. If r is constant, then this algorithm is optimal, and can be used to remove randomization from most of the "random-sampling"-based algorithms involving lines or segments [AS], [EGS], [GOS], [HW]. However, it is not efficient if r is large, in particular if $r = n^\alpha$, for some $0 < \alpha < 1$. Let us be more specific concerning this efficiency issue. Even though Matoušek's algorithm is much faster than that of Chazelle and Friedman, it is still inefficient when compared with the following "lower bound." Since the algorithm produces $O(r^2)$

triangles and each is met by $O(n/r)$ lines, the total input size of all the corresponding subproblems (that is, the total number of line–triangle crossings) is $O(nr)$ (we can show that the bound is tight in the worst case). Thus it is natural to seek an improved algorithm whose complexity is close to $O(nr)$. As we will see later, this improvement does make a difference, because in many applications we do want to choose r to be n^{α}, for some $0 < \alpha < 1$.

In this paper we achieve such an improvement, and obtain an $O(nr \log n \log^{\omega} r)$ deterministic algorithm which, given a set $\mathscr{L}$ of n lines, partitions the plane into $O(r^2)$ triangles, none of which meets more than $O(n/r)$ lines (here ω is some constant < 3.33). Then we apply this algorithm to remove randomization and to improve the time complexity of solutions to several problems. A common characteristic of the time complexity of previously known algorithms for these problems is that they are worse by a factor of $O(n^{\delta})$, for any $\delta > 0$ (with a constant of proportionality depending on δ), than the worst-case output size, or than some other natural measure of complexity that can be associated with the problem. For example, the running time of the best-known (randomized) algorithm to compute incidences between m points and n lines is $O(m^{2/3-\delta}n^{2/3+2\delta} + (m+n)\log n)$, for any $\delta > 0$ [EGS], which is worse by a factor of $O((n^2/m)^{\delta})$ than the maximum possible number of such incidences. (A slightly faster but still randomized algorithm for this problem has been given in [EGH*] which runs in $O(m^{2/3}n^{2/3}\log^4 n + (m + n^{3/2})\log^2 n)$ expected time. The difference between these two algorithms is that the first algorithm can be made deterministic without increasing its running time, using Matoušek's algorithm, while the second algorithm is not yet known to admit a similarly efficient determinization.) Factors like $O(n^{\delta})$ appear in the bound of these algorithms due to the fact that they use small values of r (in most cases constant values). Although Matoušek's algorithm makes most of these algorithms deterministic, it also cannot choose a large value of r, because its running time is quadratic in terms of r, in which case the overhead of constructing the partition dominates substantially the time complexity, resulting in an inefficient algorithm. On the other hand, by applying our algorithm, we can use a sufficiently large value of r, which allows us to remove the $O(n^{\delta})$ factor from the bounds. Another disadvantage of using a small value of r is that the algorithm then becomes recursive and more complex. In contrast, by choosing an appropriate large value of r, we partition the plane just once, because then the subproblems are sufficiently small, and can be solved directly by other means; this makes the algorithms much simpler.

In a companion paper [A] we demonstrate the usefulness of our partitioning algorithm by obtaining efficient algorithms for a variety of problems involving lines or line segments in the plane. These algorithms are deterministic, faster than previously known algorithms, and optimal up to a polylog factor in many cases. The problems that benefit from it include computing incidences between points and lines, computing many faces in arrangements of lines or segments, counting intersections in a set of segments, implicit point location, and spanning trees with low stabbing number.

Our algorithm for partitioning the plane works in two phases. The first phase partitions the plane into $O(r^2 \log^{\omega} r)$ triangles, each of which intersects $O(n/r)$ lines. The second phase reduces the number of triangles to $O(r^2)$ still maintaining the property that no triangle meets more than $O(n/r)$ lines, using a similar technique to that of Matoušek [Ma]. We believe that the second phase is unnecessary, i.e., that the first phase or some appropriate variant of it produces only $O(r^2)$ triangles, but we have not been able to prove it so far. Even if this is not the case, in most of the applications we can stop after the first phase and solve the subproblems directly within each of the resulting $O(r^2 \log^{\omega} r)$ triangles, without increasing the asymptotic complexity of the algorithm.

This paper is organized as follows. In Section 2 we describe the geometric concepts that we use. Section 3 gives an algorithm for a subproblem which is interesting in its own right, namely that of computing the κth leftmost intersection point induced by a set of n lines inside a given convex quadrilateral. Section 4 is the heart of our algorithm; it presents an algorithm to partition a given convex quadrilateral $\mathcal{Q}$ intersecting m lines and containing K of their intersection points into $O(m/\zeta + K/\zeta^2)$ convex quadrilaterals (for an arbitrary parameter ζ) so that both the number of lines crossing any subquadrilateral $\mathcal{Q}'$ and the number of their intersections inside $\mathcal{Q}'$ are small (the precise conditions are given in Theorem 4.15). In Section 5 we describe the first phase of the partitioning algorithm, which basically consists of applying the algorithm of Section 4 recursively, and Section 6 describes the second phase of the algorithm. Section 7 shows how to modify the analysis to handle degenerate cases as well. In Section 8 we conclude with some final remarks and open problems.

2. Geometric Preliminaries

This section defines the geometric concepts, and formalizes the notation that we use in this paper. Let $\mathcal{H} = \{l_1, l_2, \ldots, l_n\}$ denote a set of n lines in the plane. These lines induce a planar map called the *arrangement* $\mathcal{A}(\mathcal{H})$ of $\mathcal{H}$, whose *vertices* are the intersection points of lines in $\mathcal{H}$, *edges* are maximal connected portions of lines in $\mathcal{H}$ not containing a vertex, and *faces* are maximal connected portions of the plane not meeting any edge or vertex of $\mathcal{A}(\mathcal{H})$. See [E] for more details. For simplicity we assume that the lines in $\mathcal{H}$ are in general position, that is, no three lines are concurrent, and that no line in $\mathcal{H}$ is vertical. For any point $p \in \mathbb{R}^2$, we define its *level* to be the number of lines in $\mathcal{H}$ lying above it (not counting the lines passing through p). For any $0 \le k < n$, the *k-level* of $\mathcal{A}(\mathcal{H})$ is the set of (closure of) edges of $\mathcal{A}(\mathcal{H})$ whose level is k (see Fig. 1). A k-level of $\mathcal{A}(\mathcal{H})$ is a monotone

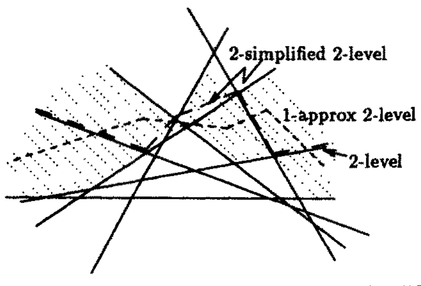

Fig. 1. The 2-level, 1-approximate 2-level, and 2-simplified 2-level.

polygonal chain with two unbounded rays. More details on k-levels can be found in [E], [EW], and [We].

Lemma 2.1. *Let p_1 and p_2 be two points in the plane whose levels are k_1 and k_2, respectively. Then the line segment $\overline{p_1p_2}$ intersects at least $|k_1 - k_2|$ lines of $\mathscr{H}$.*

Proof. Obvious from the definition of k-levels. □

Lemma 2.2 [Ma]. *Let $\mathscr{U}_1, \mathscr{U}_2, \ldots, \mathscr{U}_s$ denote pairwise disjoint sets of levels of $\mathscr{A}(\mathscr{H})$ with $|\mathscr{U}_i| \geq \varphi$, for all $i \leq s$, then there exists a level $U_i \in \mathscr{U}_i$ such that $\sum_{i=1}^{s} |U_i| \leq n^2/\varphi$.*

We call an x-monotone polygonal path Π (not necessarily formed by the edges of $\mathscr{A}(\mathscr{H})$) an *ε-approximate k-level*, for $\varepsilon \leq k$, if it lies in the strip bounded between the $k - \varepsilon$ and $k + \varepsilon$ levels of $\mathscr{A}(\mathscr{H})$ (see Fig. 1). A set of ε-approximate $2\varepsilon i$-levels, for $i \leq \lfloor n/2\varepsilon \rfloor$, is called an *$\varepsilon$-approximate leveling* of $\mathscr{A}(\mathscr{H})$. Let $p_0, p_1, \ldots, p_m$ be the vertices of a k-level of $\mathscr{A}(\mathscr{H})$. For any $\delta < m$, we define the *δ-simplified k-level* to be the polygonal path formed by connecting p_0 to p_δ, p_δ to $p_{2\delta}, \ldots, p_{\lfloor m/\delta \rfloor \delta}$ to p_m, and concatenated with the left and right rays of the k-level incident to p_0 and p_m, respectively. Edelsbrunner and Welzl [EW] proved that

Lemma 2.3 [EW]. *For a given set of n lines $\mathscr{H}$, and $0 \leq k < n$, a δ-simplified k-level is a $\lceil \delta/2 \rceil$-approximate k-level of $\mathscr{A}(\mathscr{H})$.*

Another geometric concept that we use in this paper is *duality* (see [E]. In $\mathbb{R}^2$, the dual of a line is a point, and the dual of a point is a line. The duality transformation can be chosen in such a way that it preserves the "above-below" relationship between points and lines. The dual of a line segment $\overline{pq}$ is a double wedge formed between the dual lines p^*, q^* of p, q, respectively, and not containing the vertical line through their intersection point. We denote the dual of a feature (point, line, or segment) γ by γ^* (see Fig. 2).

Let $\mathscr{Q}$ be a convex quadrilateral with vertices q_1, q_2, q_3, q_4 ordered in counterclockwise direction, and let $\partial\mathscr{Q}$ denote the boundary of $\mathscr{Q}$. Let $\mathscr{L} = \{\ell_1, \ell_2, \ldots, \ell_m\}$ denote a set of lines passing through $\mathscr{Q}$, and let K be the number of intersection points of $\mathscr{L}$ contained inside $\mathscr{Q}$. The intersection points of $\partial\mathscr{Q}$ and $\mathscr{L}$ are called the *endpoints* of $\mathscr{L}$. For simplicity let us assume that every line of $\mathscr{L}$ intersects $\partial\mathscr{Q}$ at two points, i.e., does not touch $\mathscr{Q}$ at one of its vertices (or at an edge).

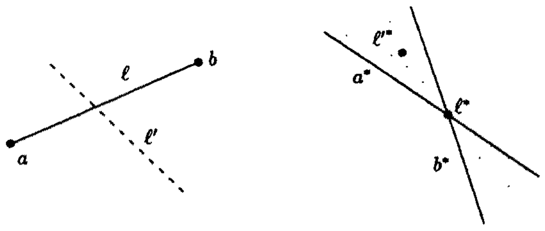

Fig. 2. A segment $e = \overline{ab}$ and its dual e^*.

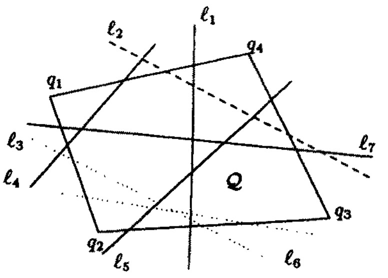

Fig. 3. Quadrilateral $\mathscr{Q}$ and set $\mathscr{L}$: solid lines denote red lines, dashed lines denote green lines, and dotted lines denote blue lines.

The set $\mathscr{L}$ of lines can be partitioned into three different classes according to the location of their endpoints (see Fig. 3):

(i) "Red" lines $\mathscr{L}_r$: a line ℓ belongs to $\mathscr{L}_r$ if one of its endpoints lies on $\overline{q_1q_2}$ or $\overline{q_2q_3}$ (referred to as the left endpoint) and the other one lies on $\overline{q_1q_4}$ *or* $\overline{q_3q_4}$ (referred to as the right endpoint), i.e., ℓ crosses the diagonal $\overline{q_1q_3}$; let $|\mathscr{L}_r| = m_r$.

(ii) "Blue" lines $\mathscr{L}_b$: a line ℓ belongs to $\mathscr{L}_b$ if one of its endpoints lies on q_1q_2 (referred to as the left endpoint) and the other one lies on $\overline{q_2q_3}$ (referred to as the right endpoint); let $|\mathscr{L}_b| = m_b$.

(iii) "Green" lines $\mathscr{L}_g$: a line ℓ belongs to $\mathscr{L}_g$ if one of its endpoints lies on $\overline{q_1q_4}$ (referred to as the left endpoint) and the other one lies on $\overline{q_3q_4}$ (referred to as the right endpoint); let $|\mathscr{L}_g| = m_g$.

These three sets are pairwise disjoint and $\mathscr{L}_r \cup \mathscr{L}_g \cup \mathscr{L}_b = \mathscr{L}$. Let K_{xy} denote the number of intersection points between $\mathscr{L}_x$ and $\mathscr{L}_y$ lying inside $\mathscr{Q}$, where $x, y \in \{r, b, g\}$. It is easily seen that $\mathscr{L}_b$ and $\mathscr{L}_g$ do not intersect inside $\mathscr{Q}$, that is $K_{bg} = 0$. Without loss of generality we can assume that $m_r \geq m/2$ because otherwise we can cyclically renumber the vertices of $\mathscr{Q}$ so that the above inequality is satisfied. For the sake of exposition we assume that q_1 is the upper left corner of $\mathscr{Q}$.

Let $A = \{a_1, \ldots, a_{m_r}\}$ (resp. $B = \{b_1, \ldots, b_{m_r}\}$) be the left (resp. right) endpoints of $\mathscr{L}_r$ appearing in counterclockwise (resp. clockwise) direction around $\partial\mathscr{Q}$. Let π, σ be the permutations defined by left and right endpoints of $\mathscr{L}_r$, respectively, that is, for a line $\ell_i \in \mathscr{L}_r$, its left (resp. right) endpoint is denoted by $a_{\pi(i)}$ (resp. $b_{\sigma(i)}$).

3. Selecting the κth Leftmost Intersection Point

Let $p_1, \ldots, p_K$ denote the intersection points of the lines in $\mathscr{L}$ lying inside $\mathscr{Q}$ and ordered in increasing x-coordinate. In this section we consider the problem of calculating the κth leftmost point p_κ of these intersections, for any given $\kappa \leq K$. The algorithm extends the recent algorithm of Cole *et al.* [CSSS] for calculating the kth leftmost intersection point of n lines in the entire plane. One of the key steps of the algorithm is to count the number of instersection points of $\mathscr{L}$ lying inside a given convex quadrilateral. We first give an algorithm that counts the number of intersections inside a convex quadrilateral in time $O(m \log m)$, where m is the number of lines passing through $\mathcal{Q}$.

3.1. *Counting Intersection Points Inside a Convex Quadrilateral*

Let $\mathcal{E}$ denote the sequence of endpoints of lines in $\mathcal{L}$ sorted along $\partial\mathcal{Q}$ in the counterclockwise direction, starting from one of its vertices. Let $a, b \in \mathcal{E}$ (with $a < b$) denote the endpoints of a line $\ell \in \mathcal{L}$. It is easily seen that ℓ intersects another line ℓ' inside $\mathcal{Q}$ if and only if exactly one of the endpoints of ℓ' lies between a and b (see Fig. 4). Therefore to count the number of intersection points inside $\mathcal{Q}$ we scan $\mathcal{E}$ once and do the following at each point of $\mathcal{E}$. If we encounter a line ℓ for the first time, we insert ℓ on top of a stack maintained as a binary tree $\mathcal{B}$, and if we encounter ℓ for the second time, we remove it from $\mathcal{B}$, but before doing so we count the number of lines in the tree that were inserted after ℓ. It can be easily verified that no intersection is counted twice. Since each operation requires $O(\log m)$ time, we have

Lemma 3.1. *For a given convex quadrilateral $\mathcal{Q}$ and a set $\mathcal{L}$ of m lines passing through it, we can count the number of intersections of $\mathcal{L}$ lying inside $\mathcal{Q}$ in time $O(m \log m)$.*

Remark 3.2. The above algorithm can count the number of intersections lying inside any k-gon $\mathcal{B}$ in $O(m \log m)$ time.

Next we give an algorithm for finding the κth leftmost intersection point inside $\mathcal{Q}$ using the counting procedure just described.

3.2. *Computing the κth Leftmost Intersection*

Cole *et al.* [CSSS] have given an $O(m \log m)$ algorithm that returns the κth leftmost intersection point in a set $\mathcal{L}$ of m lines in the plane. Their algorithm cannot be applied directly to our case because it counts all intersection points of $\mathcal{L}$ lying to the left of the κth one, while we count only those intersection points that lie inside $\mathcal{Q}$. However, we will show that the algorithm of [CSSS] can be easily adapted to our case. We assume that the reader is familiar with this algorithm, so we describe it

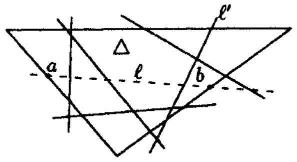

Fig. 4. A line with endpoints a and b.

here only briefly, and show how to modify it for our case; for more details, see [CSSS].

Let $\pi(a)$ denote the permutation of lines in $\mathscr{L}$ sorted in decreasing order of their intercepts with the line $x = a$, assuming that no two lines in $\mathscr{L}$ intersect at $x = a$. The *rank* of a, denoted $\psi(a)$, is defined to be the number of intersection points of $\mathscr{L}$ lying inside $\mathscr{Q}$ and to the left of $x = a$. The κth leftmost intersection point p_κ is such that $\psi(a^*) = \kappa$, where $a^* = x(p_\kappa) + \varepsilon$, for some sufficiently small $\varepsilon > 0$ (in this description we assume that no two points p_i, p_j have the same x-coordinate).

Cole *et al.*'s algorithm [CSSS] is based on Megiddo's technique [Me], further improved by Cole [Co], for implicit parametric searching. Using a sorting network, e.g., that of Ajtai *et al.* [AKS], $\pi(a^*)$ can be obtained by performing $O(m \log m)$ comparisons, performed in $O(\log m)$ batches of $O(m)$ independent comparisons each, where each comparison answers a question of the form: "Does ℓ_i lie above ℓ_j at $x = a^*$?" Although we do not know a^*, we can answer a question of this form by computing the rank of the abscissa $x(q_{ij})$, where q_{ij} is the intersection point of ℓ_i and ℓ_j. If $\psi(x(q_{ij})) > \kappa$, then q_{ij} lies to the right of $x = a^*$, so the order of ℓ_i and ℓ_j there coincides with their order at $x = -\infty$; otherwise the order is reversed. In such a sorting network there are $O(\log m)$ levels and at each level $s = O(m)$ questions are asked; let $z_1, z_2, \ldots, z_s$ denote the x-coordinates of the intersection points corresponding to the questions to be answered at a level j. If we resolve the question for the median value z_{med} of $z_1, \ldots, z_s$, then we can resolve the question for at least half of the above points, e.g., if $z_{\text{med}} < a^*$, then $z_i < a^*$ for all $z_i < z_{\text{med}}$. For the remaining half of the points, the question can be resolved by applying the same method recursively, thus allowing us to resolve all m questions by actually computing the rank of only $O(\log m)$ points. Once all comparisons are resolved at level j, we can continue to level $j + 1$ of the network. At the end, all comparisons have been resolved and we have obtained the sorted order of the lines in $\mathscr{L}$ along $x = a^*$, still without knowing the exact value of a^*. However, this order is easily seen to determine uniquely how many intersection points of $\mathscr{L}$ lie to the left of $x = a^*$, thus the set of these intersections, as well as the subset of these points lying inside $\mathscr{Q}$, are fully determined at the end of the algorithm. By keeping track of all inequalities of the form $z_{\text{med}} < a^*$ or $z_{\text{med}} > a^*$ we finally obtain an interval $\alpha < a^* < \beta$, and the logic of the procedure is easily seen to imply that α is the x-coordinate of p_κ. The point p_κ itself is also easy to obtain by recording, for each z_{med}, the two lines that induce it.

To resolve a comparison at $x = z_{\text{med}}$, we proceed as follows. Let $\mathscr{Q}^-$ denote the portion of $\mathscr{Q}$ lying to the left of the vertical line $x = z_{\text{med}}$, by Lemma 3.1 and the subsequent Remark, we can count the number of intersection points of $\mathscr{L}$ lying inside $\mathscr{Q}^-$ (which is at most a pentagon) in $O(m \log m)$ time. Thus the rank of z_{med} can be computed in $O(m \log m)$ time, so the overall running time of the algorithm is $O(m \log^3 m)$. By using Cole's improvement [Co], the time can be reduced to $O(m \log^2 m)$. Cole *et al.* [CSSS] observed that it is not necessary to compute the exact rank of a point because we are only interested in knowing whether z_{med} lies to the left or to the right of p_κ. They proved that it suffices to compute a.ı approximate rank of a point, which can be done in $O(m)$ amortized time, yielding an $O(m \log m)$ algorithm. Using the same idea we can also compute our version of approximate rank in amortized time $O(m)$, and therefore we obtain

Theorem 3.3. *Given a convex quadrilateral $\mathcal{Q}$, a set of m lines $\mathcal{L}$, and an integer κ, we can determine, in time $O(m \log m)$, a vertical line, so that $\mathcal{Q}$ contains exactly κ intersection points to the left of that line.*

Remark 3.4. (i) The remark following Lemma 3.1 implies that we can extend the above algorithm to any k-gon with $O(m \log m)$ running time.

(ii) The above algorithm uses the sorting network of Ajtai *et al.* [AKS], which involves a very large constant, therefore making our algorithm impractical. A possible solution to circumvent this problem is to use the much simpler Batcher's sorting network [B]; this network has depth $O(\log^2 m)$ and therefore the overall running time of the resulting algorithm becomes $O(m \log^2 m)$. However, to obtain a better asymptotic complexity, we use the former algorithm.

4. Partitioning a Convex Quadrilateral

This section describes an algorithm to partition the convex quadrilateral $\mathcal{Q}$ into convex quadrilateral subcells so that on the average only few lines pass through each cell, and the maximum number of lines passing through each cell is also low. To be more precise, for any given positive integer $\zeta < m$, we want to partition $\mathcal{Q}$ into $M = O(m/\zeta + K/\zeta^2)$ cells so that the following conditions are satisfied (where m_i is the number of lines passing through the ith cell, and K_i is the number of intersections with the ith cell):

(i) $\sum_{i=1}^{M} m_i = O(m + K/\zeta)$, $\max_i m_i = O(\sqrt{K})$,

(ii) $\sum_{i=1}^{M} K_i = K$, $\max_i K_i \leq \zeta\sqrt{K}$.

4.1. A Special Case

Before solving this general problem we consider a special case in which $\mathcal{Q}$ contains only "red" lines and we do not care how many intersection points lie in a cell. Moreover, although most cells in the construction to follow will be quadrilaterals, some could have up to six edges.

First, compute the left and right endpoints of all lines in $\mathcal{L}$ and sort them to obtain the lists A and B. Next, for all $1 \leq i < t = \lceil m/\zeta \rceil$, connect $a_{i\zeta}$ to $b_{i\zeta}$. The segments $\overline{a_{i\zeta}b_{i\zeta}}$ are called *pseudoedges* (see Fig. 5). The pseudoedges partition $\mathcal{Q}$ into cells $\mathcal{Q}_1, \mathcal{Q}_2, \ldots, \mathcal{Q}_t$. The following sequence of lemmas show that this partitioning has the desired properties.

Lemma 4.1. *A line $\ell_i \in \mathcal{L}$ intersects at least $|\pi(i) - \sigma(i)|$ lines of $\mathcal{L}$ inside $\mathcal{Q}$.*

Proof. Without loss of generality assume that $\pi(i) \geq \sigma(i)$. Since only $\sigma(i) - 1$ lines have their right endpoints preceding the right endpoint of ℓ_i, at least $\pi(i) - \sigma(i)$ of the lines, whose left endpoints precede that of ℓ_i, have their right endpoints after $b_{\sigma(i)}$ in B. But every such line crosses the line ℓ_i inside $\mathcal{Q}$, showing that at least $|\pi(i) - \sigma(i)|$ lines of $\mathcal{L}$ intersect ℓ_i inside $\mathcal{Q}$. □

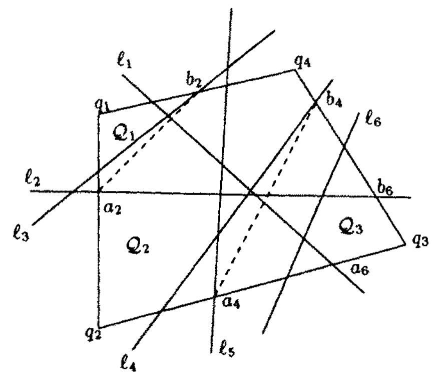

Fig. 5. Partitioning $\mathscr{Q}$ into $\mathscr{Q}_1, \ldots, \mathscr{Q}_t$ with $m = 6$ and $\zeta = 2$. The dashed edges are pseudoedges.

Let ζ_i denote the number of pseudoedges intersected by the line ℓ_i.

Lemma 4.2. $\sum_{i=1}^{m} \delta_i \leq 2K/\zeta + m$.

Proof. If a line ℓ_i intersects δ_i pseudoedges, then it is easily seen that

$$|\pi(i) - \sigma(i)| \geq (\delta_i - 1)\zeta.$$

Summing over all lines, we obtain

$$\sum_{i=1}^{m} |\pi(i) - \sigma(i)| \geq \sum_{i=1}^{m} (\delta_i - 1)\zeta.$$

Let ν_i denote the number of lines of $\mathscr{L}$ that ℓ_i intersects inside $\mathscr{Q}$. We have $\sum_{i=1}^{m} \nu_i = 2K$, because there are only K intersection points inside $\mathscr{Q}$ and each intersection is counted twice. By Lemma 4.1, $|\pi(i) - \sigma(i)| \leq \nu_i$, therefore

$$\sum_{i=1}^{m} (\delta_i - 1)\zeta \leq 2K \quad \text{or} \quad \sum_{i=1}^{m} \delta_i \leq \frac{2K}{\zeta} + m. \qquad \square$$

Let m_i denote the number of lines passing through the cell $\mathscr{Q}_i$. Since the line ℓ_j lies in $\delta_j + 1$ cells, we have

Corollary 4.3. $\sum_{i=1}^{t} m_i = \sum_{j=1}^{m} (\delta_j + 1) \leq 2(K/\zeta + m)$.

Lemma 4.4. *Let μ_i denote the number of lines passing through the cell $\mathscr{Q}_i$ but not having any of their endpoints on $\partial\mathscr{Q}_i$. Then $\mu_i \leq 2\sqrt{K}$ for all $i \leq t$.*

Proof. Let $\mathscr{L}_i$ denote the set of lines passing through the cell $\mathscr{Q}_i$ but not having any endpoint in $\mathscr{Q}_i$. Partition the set $\mathscr{L}_i$ into disjoint subsets $\mathscr{L}_{xy}$, for $x =$

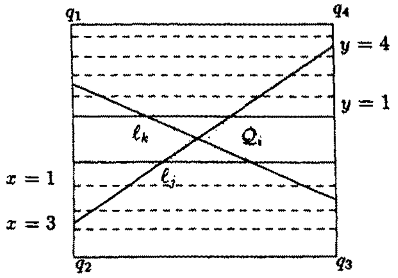

Fig. 6. Line ℓ_j intersects at least 6ζ lines of $\mathscr{L}$.

$1, \dots, t - i$, $y = 1, \dots, i - 1$, such that a line ℓ_j belongs to $\mathscr{L}_{xy}$ if one of its endpoints lies in $\mathscr{Q}_{i+x}$ and the other in $\mathscr{Q}_{i-y}$. Let $\mu_{xy} = |\mathscr{L}_{xy}|$. Note that

$$\forall \ell_j \in \mathscr{L}_{xy}, \qquad |\pi(j) - \sigma(j)| \geq \zeta(x + y - 1).$$

By Lemma 4.1, a line $\ell_j \in \mathscr{L}_{xy}$ intersects at least $(x + y - 1)\zeta$ lines (see Fig. 6). Summing over all x and y we get

$$2K \geq \sum_x \sum_y \mu_{xy}(x + y - 1) \cdot \zeta$$

or

$$\frac{2K}{\zeta} \geq \sum_x \sum_y x \cdot \mu_{xy} + \sum_x \sum_y y \cdot \mu_{xy} - \sum_x \sum_y \mu_{xy}. \tag{4.1}$$

Let $\mathscr{L}'_x \subseteq \mathscr{L}_i$ denote the set of all lines having one endpoint in $\mathscr{Q}_{i+x}$. Then

$$\sum_x \sum_y x \cdot \mu_{xy} = \sum_x x\left(\sum_x \mu_{xy}\right) = \sum_x x|\mathscr{L}'_x|. \tag{4.2}$$

Similarly, if $\mathscr{L}''_y \subseteq \mathscr{L}_i$ denotes the set of lines having one endpoint in $\mathscr{Q}_{i-y}$, then

$$\sum_x \sum_y y \cdot \mu_{xy} = \sum_y y|\mathscr{L}''_y|.$$

Moreover $\sum_x \sum_y \mu_{xy} = \mu_i$. Therefore (4.1) can be written as

$$\sum_x x \cdot |\mathscr{L}'_x| + \sum_y y \cdot |\mathscr{L}''_y| - \mu_i \leq \frac{2K}{\zeta}. \tag{4.3}$$

Since $\sum_x |\mathscr{L}'_x| = \mu_i$ and, for all x, $|\mathscr{L}'_x| \leq 2\zeta$, the term $\sum_x x|\mathscr{L}'_x|$ minimizes when

$$|\mathscr{L}'_x| = \begin{cases} 2\zeta & \text{for} \quad 1 \leq x \leq \lfloor \mu_i/2\zeta \rfloor, \\ \mu_i - 2\zeta\left\lfloor \dfrac{\mu_i}{2\zeta} \right\rfloor & \text{for} \quad x = \lfloor \mu_i/2\zeta \rfloor + 1, \\ 0 & \text{otherwise.} \end{cases}$$

Hence,

$$\sum_x x\cdot|\mathscr{L}'_x| \ge \sum_{x=1}^{\lfloor \mu_i/2\zeta \rfloor} x\cdot 2\zeta + \left(\mu_i - 2\zeta\left\lfloor \frac{\mu_i}{2\zeta}\right\rfloor\right)\left(\left\lfloor\frac{\mu_i}{2\zeta}\right\rfloor + 1\right)$$
$$\ge \left(\left\lfloor\frac{\mu_i}{2\zeta}\right\rfloor + 1\right)\left(\mu_i - \zeta\left\lfloor\frac{\mu_i}{2\zeta}\right\rfloor\right). \tag{4.4}$$

Similarly,

$$\sum_y y\cdot|\mathscr{L}''_y| \ge \left(\left\lfloor\frac{\mu_i}{2\zeta}\right\rfloor + 1\right)\left(\mu_i - \zeta\left\lfloor\frac{\mu_i}{2\zeta}\right\rfloor\right). \tag{4.5}$$

By substituting (4.4) and (4.5) in (4.3), we obtain

$$\frac{2K}{\zeta} \ge 2\left(\left\lfloor\frac{\mu_i}{2\zeta}\right\rfloor + 1\right)\left(\mu_i - \zeta\left\lfloor\frac{\mu_i}{2\zeta}\right\rfloor\right) - \mu_i. \tag{4.6}$$

Let $\lfloor \mu_i/2\zeta \rfloor = a$ and let $\mu_i/2\zeta = a + v$, for some $0 \le v < 1$, then (4.6) becomes

$$\frac{2K}{\zeta} \ge 2(a+1)(2\zeta(a+v) - \zeta a) - 2\zeta(a+v)$$

or

$$\begin{aligned}\frac{K}{\zeta^2} &\ge (a+1)(a+2v) - (a+v)\\ &= (a+v)^2 + v(1-v)\\ &\ge (a+v)^2 \qquad \text{(because } 0 \le v < 1)\\ &= \frac{\mu_i^2}{4\zeta^2}.\end{aligned}$$

Therefore, $\mu_i \le 2\sqrt{K}$. □

Concerning running time, notice that the sets A and B, and therefore $\mathscr{Q}_1, \ldots, \mathscr{Q}_t$, can be computed in time $O(m \log m)$. Moreover, for each line ℓ_j, we can easily compute the cells it crosses—they are simply all the cells lying between (and including) the cell containing $a_{\pi(j)}$ and the cell containing $b_{\sigma(j)}$. Thus the total time spent in computing the lines passing through each cell is bounded by $O(m \log m + K/\zeta)$. Now it follows from Lemma 3.1 that number K_i of intersections within $\mathscr{Q}_i$ can be computed in time $O(m_i \log m_i)$. Therefore the total time spent in computing intersections is at most

$$O\left(\sum_i m_i \log m_i\right) = O\left(\left(\sum_i m_i\right)\log m\right) = O\left(\left(m + \frac{K}{\zeta}\right)\log m\right).$$

Hence, we can conclude that

Lemma 4.5. *The above algorithm partitions $\mathcal{Q}$ into $t = \lceil m/\zeta \rceil$ subcells $\mathcal{Q}_1, \dots, \mathcal{Q}_t$ in time $O(m \log m)$ such that*

(i) $\sum_{i=1}^{t} m_i \le 2(m + K/\zeta)$, *and*

(ii) $\max_i m_i \le 2\sqrt{K} + 2\zeta$.

Moreover, we can compute $\mathcal{L}_i$ and K_i (as defined above) for all $\mathcal{Q}_i$, in time $O((m + K/\zeta) \log m)$.

Remark 4.6. Note that in the special case at hand we have ignored the issue of making the numbers K_i small. Also, the resulting partitioning can have one cell with six edges or two cells with five edges each. These issues will be addressed in the following general algorithm.

4.2. General Algorithm

Next we give an algorithm for the general case using the above procedure as a subroutine. Our algorithm consists of five steps.

The first step applies the preceding algorithm to the collection of red lines, to partition the quadrilateral $\mathcal{Q}$ into a collection of cells $\mathbf{Q}^{\mathrm{r}} = \{\mathcal{Q}_1^{\mathrm{r}}, \dots, \mathcal{Q}_t^{\mathrm{r}}\}$, for $t = \lceil m_{\mathrm{r}}/\zeta \rceil$, so that every cell contains the left (resp. right) endpoints of at most ζ red lines. Let $R = \{r_1, \dots, r_{t-1}\}$ denote the set of pseudoedges that bound the quadrilaterals of $\mathbf{Q}^{\mathrm{r}}$; they are referred to as "red" pseudoedges.

The second step applies the preceding algorithm to the collection of green lines (where this time $\partial\mathcal{Q}$ is divided into a left and a right portion at q_4 and q_2), to partition $\mathcal{Q}$ into a collection of cells $\mathbf{Q}^{\mathrm{g}} = \{\mathcal{Q}_1^{\mathrm{g}}, \dots, \mathcal{Q}_u^{\mathrm{g}}\}$, for $u = \lceil m_g/\zeta \rceil$, so that every cell contains the left (resp. right) endpoints of at most ζ green lines. Let $G = \{g_1, \dots, g_{u-1}\}$ denote the set of pseudoedges bounding the quadrilaterals of $\mathbf{Q}^{\mathrm{g}}$; these pseudoedges are referred to as "green" pseudoedges. Since the endpoints of all green lines lie on $\overline{q_1q_4}$ and $\overline{q_3q_4}$, all green pseudoedges extend from $\overline{q_1q_4}$ to $\overline{q_3q_4}$. Assume that the quadrilaterals $\mathcal{Q}_i^{\mathrm{g}}$ are sorted in their order along $\overline{q_1q_4}$, and that $\mathcal{Q}_i^{\mathrm{g}}$ is bounded by the two pseudoedges g_{i-1}, g_i, for $1 < i < u$.

In the third step we apply the same algorithm to the collection of blue lines (again, partitioning $\partial\mathcal{Q}$ at q_2 and q_4), to partition $\mathcal{Q}$ into a collection of cells $\mathbf{Q}^{\mathrm{b}} = \{\mathcal{Q}_1^{\mathrm{b}}, \dots, \mathcal{Q}_v^{\mathrm{b}}\}$, for $v = \lceil m_{\mathrm{b}}/\zeta \rceil$, so that every cell contains the left (resp. right) endpoints of at most ζ blue lines. Let $B = \{b_1, \dots, b_{v-1}\}$ denote the set of pseudoedges bounding the quadrilaterals of $\mathbf{Q}^{\mathrm{b}}$; they are referred to as "blue" pseudoedges. This time all pseudoedges extend from $\overline{q_1q_2}$ to $\overline{q_2q_3}$. Assume that the quadrilaterals $\mathcal{Q}_i^{\mathrm{b}}$ are sorted in their order along $\overline{q_1q_2}$, and that $\mathcal{Q}_i^{\mathrm{b}}$ is bounded by the two pseudoedges b_{i-1}, b_i, for $1 < i < v$. See Fig. 7 for an illustration of the resulting pseudoedges.

Let $\mathcal{S} = R \cup B \cup G$, and let $\mathcal{E}$ denote the set of endpoints of segments in $\mathcal{S}$. Let $\mathbf{F}$ denote the set of bounded faces in the arrangement of $\partial\mathcal{Q} \cup \mathcal{S}$. If a face $\mathcal{F}_\alpha \in \mathbf{F}$ touches the boundary of $\mathcal{Q}$, then it is called a *boundary cell*, otherwise it is called an

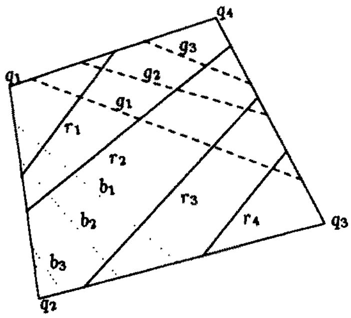

Fig. 7. Red, green, and blue pseudoedges.

internal cell (see Fig. 8). The boundary of an internal cell is formed by four pseudoedges, two of which are always red. On the basis of the color of their edges we partition the cells into three catagories:

(i) Red-green cells: cells formed by the intersection of $\mathscr{Q}_i^g$ for $1 < i \leq u$ and $\mathscr{Q}_j^r$ for $1 \leq j \leq t$. A red-green cell formed by $\mathscr{Q}_i^g \cap \mathscr{Q}_j^r$ is denoted by $\mathscr{F}_{ij}$. If $\mathscr{F}_{ij}$ is an internal cell, then its edges are portions of $g_{i-1}, g_i \in G$, and $r_{j-1}, r_j \in R$ (and $i < u$, $1 < j < t$).

(ii) Red-blue cells: cells formed by the intersection of $\mathscr{Q}_i^b$ for $1 < i \leq v$ and $\mathscr{Q}_j^r$ for $1 \leq j \leq t$. A red-blue cell formed by $\mathscr{Q}_i^b \cap \mathscr{Q}_j^r$ is denoted by $\mathscr{F}_{ij}$. If $\mathscr{F}_{ij}$ is an internal cell, then its edges are portions of $b_{i-1}, b_i \in B$, and $r_{j-1}, r_j \in R$ (and $i < v$, $1 < j < t$).

(iii) Red-blue-green cells: cells formed by the intersection of $\mathscr{Q}_1^g$, $\mathscr{Q}_1^b$, and $\mathscr{Q}_i^r$ for $1 \leq i \leq t$. We denote such a cell by $\mathscr{F}_{1i}$. If $\mathscr{F}_{1i}$ is an internal cell, then its edges are portions of b_1, g_1, r_{i-1}, and r_i (and $1 < i < t$).

The problem with the boundary cells produced by the arrangement of $\partial\mathscr{Q} \cup \mathscr{S}$ is that some of them may have more than four edges. However, since every cell has at most four pseudoedges, it cannot have more then eight edges in total. At the fourth step of our algorithm, we partition all boundary cells with more than four edges into two or three convex quadrilaterals and triangles by appropriate diagonals, which we also call pseudoedges (see Fig. 9).

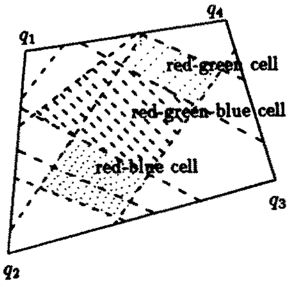

Fig. 8. Internal and boundary cells: shaded regions denote internal cells.

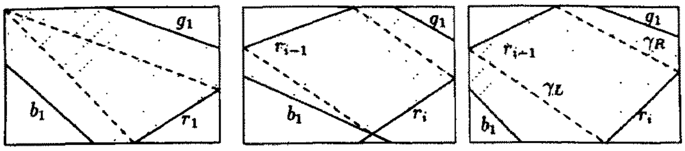

Fig. 9. Cells with seven or eight edges: dashed lines are the pseudoedges added to partition the cell into convex quadrilaterals and triangles.

A cell with five or six edges is partitioned arbitrarily into two convex quadrilaterals, but a cell with seven or eight edges is partitioned in such a way so that one of the new pseudoedges intersects only red and green lines, and the other intersects only red and blue lines. It is easily seen that such a partition is always possible. For example, an 8-cell lying between the red pseudoedges r_{i-1}, r_i is partitioned by connecting the left endpoint of r_{i-1} to the left endpoint of r_i, and the right endpoint of γ_{i-1} to the right endpoint of r_i (see Fig. 9). 7-cells are partitioned in a similar manner.

Note that there can be at most one cell having seven or eight edges. The existence of an 8-cell implies that no pair of pseudoedges intersect, and the existence of a 7-cell allows only very limited pattern of pseudoedge intersections. Moreover the existence of a 7- or 8-cell implies that the first step creates at least one red pseudoedge.

Next, for each of the cells produced so far, we compute the line passing through it, and count the number of intersection points lying in it. If $K \geq 6\zeta^2$ and a cell $\mathscr{F}_\alpha$ has $K_\alpha > \zeta\sqrt{K}$ intersection points, then we partition it into $\lceil K_\alpha/\zeta\sqrt{K}\rceil$ cells, each containing at most $\zeta\sqrt{K}$ intersections, using the algorithm of Section 3. With some care we can obtain such a partition with at most one 5-cell. Every 5-cell is further partitioned by a diagonal into two subcells. This yields a final collection of quadrilaterals $\{\mathscr{Q}_1, \mathscr{Q}_2, \ldots, \mathscr{Q}_M\}$, which is the output of the algorithm.

4.3. *Analysis of the Algorithm*

The algorithm just described for partitioning a convex quadrilateral is fairly simple. Its analysis, however, is not. For $i = 1, \ldots, M$, let m_i denote the number of lines passing through $\mathscr{Q}_i$, and let K_i denote the number of intersections contained in $\mathscr{Q}_i$. For simplicity we assume that the lines are in general position, that is, no three lines are concurrent, and no line is vertical. We begin by bounding the total number of cells created by the algorithm and the number of lines meeting each cell. As we shall see later, the constants appearing in the bounds for $\max_i m_i$ and $\sum_i m_i$ control the exponent of the logarithmic factor in the bound for the time complexity of the overall partitioning algorithm, therefore we try to obtain as small constants as possible. If we do not worry about constants, the proof can be simplified a lot. To bound the total number of cells produced by our algorithm, we first estimate the number of cells formed by overlapping the original red, green, and blue pseudoedges.

Lemma 4.7. *Let N be the number of cells produced by overlapping the red, green, and blue pseudoedges, then*

$$N \leq \frac{2}{\zeta} m + \frac{K}{\zeta^2} - 2. \tag{4.7}$$

Proof. We bound the number of boundary and internal cells separately. The number of boundary cells is obviously equal to the number of endpoints of pseudoedges, namely $2|\mathscr{S}| = 2(u + v + t - 3)$.

Next, consider the number of internal cells. To bound the number of red-green and red-blue-green cells, we use the following charging scheme. Let w be the intersection point of a red line ℓ and a green line ℓ'. Suppose that the right endpoint of ℓ lies on $\overline{q_1 q_4}$ in a cell $\mathscr{Q}_j^{\mathrm{r}}$, and that the left endpoint of ℓ' (on $\overline{q_1 q_4}$) lies in $\mathscr{Q}_i^{\mathrm{g}}$. If $i = 1$ and $1 < j < t$, then we charge w to the red-blue-green cell $\mathscr{F}_{1j}$ (e.g., w_4 in Fig. 10), and if $1 < i < u$ and $1 < j < t$, then we charge w to the red-green cell $\mathscr{F}_{ij}$ (e.g., w_2 in Fig. 10). If $\mathscr{Q}_i^{\mathrm{g}} \cap \mathscr{Q}_j^{\mathrm{r}}$ is a boundary cell, then we do not charge w to any cell (e.g., w_3 in Fig. 10). Similarly, if the right endpoint of ℓ lies on $\overline{q_3 q_4}$, then we charge w to the cell $\mathscr{F}_{ij}$ (e.g., w_1 in Fig. 10), where $\mathscr{Q}_i^{\mathrm{g}}$ contains the right endpoint of ℓ' (on $\overline{q_3 q_4}$) and $\mathscr{Q}_j^{\mathrm{r}}$ contains, as before, the right endpoint of ℓ. Again, no charge is made if $\mathscr{Q}_i^{\mathrm{g}} \cap \mathscr{Q}_j^{\mathrm{r}}$ is a boundary cell.

Clearly, each internal red-green intersection is charged to at most one cell. Morever, it is easily checked that each internal red-green or red-blue-green cell is charged exactly ζ^2 red-green intersections, namely those between the red lines whose right endpoint lies in $\mathscr{Q}_j^{\mathrm{r}}$ and the green lines whose right or left endpoint (as the case might be) lies in $\mathscr{Q}_i^{\mathrm{g}}$ (by construction, all those intersections points do lie inside $\mathscr{Q}$). Hence there are at most K_{rg}/ζ^2 red-green and red-blue-green cells. Similarly, we can prove that there are at most K_{rb}/ζ^2 red-blue cells. Thus, the overall number N of cells after the third step is bounded by

$$\begin{aligned} N &\leq 2(t + u + v) + \frac{K_{\mathrm{rg}}}{\zeta^2} + \frac{K_{\mathrm{rb}}}{\zeta^2} - 6 \\ &\leq \left(\left\lceil \frac{m_{\mathrm{r}}}{\zeta} \right\rceil + \left\lceil \frac{m_{\mathrm{g}}}{\zeta} \right\rceil + \left\lceil \frac{m_{\mathrm{b}}}{\zeta} \right\rceil\right) + \frac{1}{\zeta^2}(K_{\mathrm{rg}} + K_{\mathrm{rb}}) - 6 \\ &\leq \frac{2}{\zeta}(m + 2) + \frac{1}{\zeta^2}(K_{\mathrm{rg}} + K_{\mathrm{rb}}) - 6 \\ &\leq \frac{2}{\zeta} m + \frac{K}{\zeta^2} - 2. \end{aligned}$$ □

Lemma 4.8. *Let M be the total number of cells produced by the above algorithm. Then*

$$M \leq \frac{3}{\zeta} m + \frac{2K}{\zeta^2}.$$

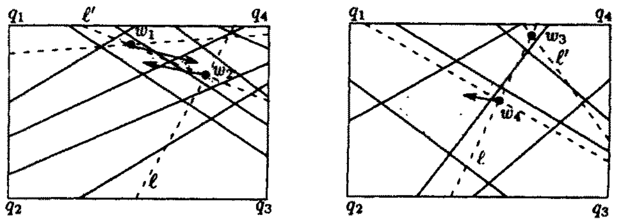

Fig. 10. Charging of a red-green intersection point.

Proof. It is easy to see that for any two adjacent boundary cells, one of them has at most four edges, therefore there are at most $|\mathcal{S}|$ cells with five or more edges (see Fig. 11). Moreover, there is at most one cell with seven or eight edges. Since each 5- or 6-cell is partitioned into two convex quadrilaterals and the 7- or 8-cell into three convex quadrilaterals, the fourth step creates at most $|\mathcal{S}| + 1$ convex quadrilaterals, which in conjunction with (4.7) implies that after the fourth step the number of cells is at most

$$\frac{2m}{\zeta} + \frac{K}{\zeta^2} - 2 + (t + u + v - 3) \le \frac{3m}{\zeta} + \frac{K}{\zeta^2} - 5.$$

Finally, the fifth step adds at most $\lceil K_\alpha/\zeta\sqrt{K}\rceil$ new cells at each quadrilateral containing $K_\alpha > \zeta\sqrt{K}$ intersection points. Since there can be at most $K/(\zeta\sqrt{K})$ such quadrilaterals, the total number of new cells is at most

$$\sum_\alpha \left(\frac{K_\alpha}{\zeta\sqrt{K}} + 1\right) \le \frac{2\sqrt{K}}{\zeta} \le \frac{K}{\zeta^2},$$

because we apply this step only for $K > 4\zeta^2$. Thus

$$M \le \frac{3m}{\zeta} + \frac{2K}{\zeta^2}. \qquad \Box$$

Next, we bound the maximum number of lines passing through any cell and the total number of line-cell crossings.

Lemma 4.9. *For each $1 \le i \le M$, the number m_i of lines passing through the cell $\mathcal{Q}_i$ satisfies*

$$m_i \le \max\{2\sqrt{2}\sqrt{K} + 4\zeta, 2\sqrt{K} + 6\zeta\}. \tag{4.8}$$

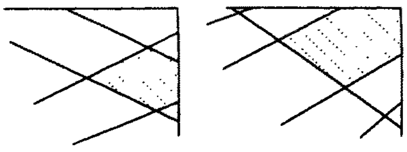

Fig. 11. No two adjacent cells have more than four edges.

Proof. Note that a cell $\mathscr{Q}_i$ produced by our algorithm is a subset of some cell $\mathscr{F}_\alpha \in \mathbf{F}$ (the cells obtained by overlapping red, blue, and green pseudoedges). Therefore it suffices to bound the number of lines passing through a cell in $\mathbf{F}$. First, let us consider a red-green cell $\mathscr{F}_{ij} = \mathscr{Q}_i^{\mathrm{g}} \cap \mathscr{Q}_j^{\mathrm{r}}$. By Lemma 4.5, $\mathscr{Q}_i^{\mathrm{g}}$ meets at most $2\sqrt{K_{\mathrm{gg}}} + 2\zeta$ green lines and $\mathscr{Q}_j^{\mathrm{r}}$ meets at most $2\sqrt{K_{\mathrm{rr}}} + 2\zeta$ red lines, and since no blue line passes through a red-green cell, there are at most $2(\sqrt{K_{\mathrm{rr}}} + \sqrt{K_{\mathrm{gg}}} + 2\zeta)$ lines passing through $\mathscr{F}_{ij}$. Similarly, we can show that a red-blue cell meets at most $2(\sqrt{K_{\mathrm{rr}}} + \sqrt{K_{\mathrm{bb}}} + 2\zeta)$ lines.

Finally, let $\mathscr{F}_{1i}$ be a red-blue-green cell, then $\mathscr{F}_{1i}$ is the intersection of $\mathscr{Q}_1^{\mathrm{g}}, \mathscr{Q}_1^{\mathrm{b}}$, *and* $\mathscr{Q}_i^{\mathrm{r}}$. Since $\mathscr{Q}_1^{\mathrm{g}}$ (resp. $\mathscr{Q}_1^{\mathrm{b}}$) meets at most 2ζ green (resp. blue) lines, $\mathscr{F}_{1i}$ has at most $2\sqrt{K_{\mathrm{rr}}} + 6\zeta$ lines. Now the lemma follows from the fact that $\max\{\sqrt{K_{\mathrm{rr}}} + \sqrt{K_{\mathrm{bb}}}, \sqrt{K_{\mathrm{rr}}} + \sqrt{K_{\mathrm{gg}}}\} \le \sqrt{2K}$. $\square$

For $x, y \in \{\mathrm{r}, \mathrm{b}, \mathrm{g}\}$, let δ_i^{xy} denote the number of pseudoedges of color x intersected by the line ℓ_i of color y. It follows from Lemma 4.2 that

$$\sum_{i=1}^{m_x} \delta_i^{xx} \le \frac{2K_{xx}}{\zeta} + m_x.$$

In the next two lemmas we bound the "mixed" sums $\sum_{i=1}^{m_{\mathrm{g}}} \delta_i^{\mathrm{rg}}, \sum_{i=1}^{m_{\mathrm{b}}} \delta_i^{\mathrm{rb}}, \sum_{i=1}^{m_{\mathrm{r}}} \delta_i^{\mathrm{gr}}$, and $\sum_{i=1}^{m_{\mathrm{r}}} \delta_i^{\mathrm{br}}$.

Lemma 4.10. $\sum_{i=1}^{m_{\mathrm{g}}} \delta_i^{\mathrm{rg}} \le K_{rg}/\zeta + m_{\mathrm{g}}$ *and* $\sum_{i=1}^{m_{\mathrm{b}}} \delta_i^{\mathrm{rb}} \le K_{\mathrm{rb}}/\zeta + m_{\mathrm{b}}$.

Proof. We only prove the first part of the lemma; the second part can be proved in a symmetric way. Let ℓ_i be a green line intersecting δ_i^{rg} red pseudoedges, $r_{k+1}, \ldots, r_{k+\delta_i^{\mathrm{rg}}}$, and let ℓ' be a red line having its right endpoint in $\mathscr{Q}_l^{\mathrm{r}}$ for $k+1 < l < k + \delta_i^{\mathrm{rg}}$ (see Fig. 12(a)). It is easily checked that ℓ' intersects ℓ_i inside $\mathscr{Q}$. Since each $\mathscr{Q}_l^{\mathrm{r}}$ contains the right endpoints of ζ lines, ℓ_i intersects at least $(\delta_i^{\mathrm{rg}} - 1)\zeta$ red lines inside $\mathscr{Q}$. But there are at most K_{rg} red-green intersection points, therefore summing over all green lines, we obtain

$$\sum_{i=1}^{m_{\mathrm{g}}} (\delta_i^{\mathrm{rg}} - 1)\zeta \le K_{\mathrm{rg}} \quad \text{or} \quad \sum_{i=1}^{m_{\mathrm{g}}} \delta_i^{\mathrm{rg}} \le \frac{K_{\mathrm{rg}}}{\zeta} + m_{\mathrm{g}}. \qquad \square$$

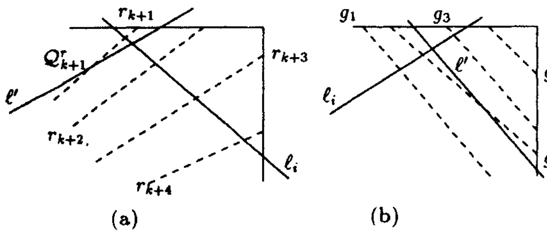

Fig. 12. Illustration for (a) Lemma 4.10 and (b) Lemma 4.11.

Lemma 4.11. $\sum_{i=1}^{m_r} \delta_i^{\mathrm{gr}} \le K_{\mathrm{rg}}/\zeta$ *and* $\sum_{i=1}^{m_r} \delta_i^{\mathrm{br}} \le K_{\mathrm{rb}}/\zeta$.

Proof. Let ℓ_i be a red line intersecting δ_i^{gr} green pseudoedges, $g_1, \ldots, g_{\delta_i^{\mathrm{gr}}}$. Suppose that the right endpoint of ℓ_i lies on $\overline{q_1 q_4}$. Let ℓ' be a green line whose left endpoint (on $\overline{q_1 q_4}$) lies in $\mathscr{Q}_l^{\mathrm{g}}$ for $l \le \delta_i^{\mathrm{gr}}$ (see Fig. 12(b)). It is easily seen that ℓ' intersects ℓ_i inside $\mathscr{Q}$. Similarly, if the right endpoint of ℓ_i is on $\overline{q_3 q_4}$, then ℓ_i intersects any green line whose right endpoint (on $\overline{q_3 q_4}$) lies in $\mathscr{Q}_l^{\mathrm{g}}$ for $l \le \delta_i^{\mathrm{gr}}$. Since each $\mathscr{Q}_l^{\mathrm{g}}$ contains ζ left endpoints, and also ζ right endpoints, of green lines, ℓ_i intersects at least $\delta_i^{\mathrm{gr}} \cdot \zeta$ green lines inside $\mathscr{Q}$. Summing over all red lines, we obtain

$$\sum_{i=1}^{m_r} \delta_i^{\mathrm{gr}} \cdot \zeta \le K_{\mathrm{rg}} \quad \text{or} \quad \sum_{i=1}^{m_r} \delta_i^{\mathrm{gr}} \le \frac{K_{\mathrm{rg}}}{\zeta}.$$

Similarly, we can prove that $\sum_{i=1}^{m_r} \delta_i^{\mathrm{br}} \le K_{\mathrm{rb}}/\zeta$. □

Lemma 4.12. *Let m_i' denote the number of lines passing through the* i*th cell of* **F**, *then*

$$\sum_{i=1}^{N} m_i' \le \frac{2}{\zeta} K + \frac{5}{2} m. \tag{4.9}$$

Proof. Let η_j^x denote the number of cells crossed by a line $\ell_j \in \mathscr{L}_x$, where $x \in \{\mathrm{r}, \mathrm{b}, \mathrm{g}\}$. Each cell $\mathscr{F}$ meeting ℓ_j is charged to the leftmost point $w_{\mathscr{F}}$ of $\partial\mathscr{F} \cap \ell_j$, therefore for all cells crossed by ℓ_j, except the leftmost one, $w_{\mathscr{F}}$ is an intersection point of a pseudoedge and ℓ_j (see Fig. 13). For red lines it follows from Lemmas 4.2 and 4.11 that

$$\sum_{i=1}^{m_r} \eta_i^{\mathrm{r}} = \sum_{i=1}^{m_r} (\delta_i^{\mathrm{rr}} + \delta_i^{\mathrm{gr}} + \delta_i^{\mathrm{br}} + 1)$$
$$\le 2\frac{K_{\mathrm{rr}}}{\zeta} + \frac{K_{\mathrm{rg}}}{\zeta} + \frac{K_{\mathrm{rb}}}{\zeta} + 2m_{\mathrm{r}}. \tag{4.10}$$

Similarly, using Lemmas 4.2 and 4.10 we can show that

$$\sum_{i=1}^{m_{\mathrm{b}}} \eta_i^{\mathrm{b}} = \sum_{i=1}^{m_{\mathrm{b}}} (\delta_i^{\mathrm{bb}} + \delta_i^{\mathrm{rb}} + \delta_i^{\mathrm{gb}} + 1) \tag{4.11}$$
$$\le 2\frac{K_{\mathrm{bb}}}{\zeta} + \frac{K_{\mathrm{rb}}}{\zeta} + 3m_{\mathrm{b}}$$

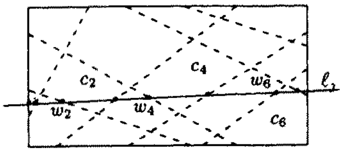

Fig. 13. Line ℓ_j and the cells in **F**; cell c_i is charged to w^i.

and

$$\sum_{i=1}^{m_g} \eta_i^g \le 2\frac{K_{gg}}{\zeta} + \frac{K_{rg}}{\zeta} + 3m_g. \tag{4.12}$$

Since $\sum_{i=1}^{N} m_i' = \sum_{x\in\{r,b,g\}} \sum_{j=1}^{m_x} \eta_j^x$, summing (4.10), (4.11), and (4.12), we obtain

$$\sum_{i=1}^{N} m_i' \le \frac{2}{\zeta}(K_{rr} + K_{bb} + K_{gg} + K_{rg} + K_{rb}) + 3(m_g + m_b) + 2m_r$$

$$\le \frac{2}{\zeta}K + \frac{5}{2}m \qquad \text{(because } m_r \ge m/2\text{; see Section 2).} \qquad \square$$

Next, we bound $\sum_{i=1}^{M} m_i$, that is, the total number of line-cell crossings, for the final cells produced by the algorithm.

Lemma 4.13. *Let m_i denote the number of lines in cell $\mathcal{Q}_i$; then*

$$\sum_{i=1}^{M} m_i \le (4 + 4\sqrt{2})\frac{K}{\zeta} + (5 + 4\sqrt{2})m.$$

Proof. Let m_α denote the number of lines meeting the αth cell after the fourth step. First consider the case when F has a cell $\mathcal{F}_\alpha$ with seven or eight edges, say eight edges (see Fig. 9). Let r_{i-1} and r_i denote the red pseudolines bounding $\mathcal{F}_\alpha$. In this case the fourth step adds only two pseudoedges, γ_L and γ_R. Since γ_L does not intersect green lines and intersects only those red lines whose left endpoints lie in $\mathcal{Q}_i^r$, it adds at most $\zeta + m_b$ line-cell crossings. Similarly γ_R adds at most $\zeta + m_g$ line-cell crossings. Moreover, if F has a 7-cell, then it can also have one 5-cell. Thus by Lemma 4.12,

$$\sum_\alpha m_\alpha \le \frac{2}{\zeta}K + \frac{5}{2}m + (2\zeta + m_g + m_b) + m.$$

Note that in this case the first three steps create at least three pseudoedges, which implies that $m > 3\zeta$. Since $m_g + m_b \le m/2$, we have

$$\sum_\alpha m_\alpha \le \frac{2}{\zeta}K + 5m.$$

On the other hand, if all cells after the third step have at most six edges, then every cell is partitioned into at most two cells. Thus by Lemma 4.12, the total number of line–cell crossing after the fourth step is

$$\sum_{\alpha} m_\alpha \le \frac{4}{\zeta} K + 5m.$$

It is easily seen that the final step creates at most $2\sqrt{K}/\zeta$ new cells. Moreover, in that step a cell $\mathscr{F}_\alpha$ is partitioned only when $K > 6\zeta^2$, in which case $m_\alpha < 2\sqrt{2}\sqrt{K} + 4\zeta$ (see Lemma 4.9). Hence, we obtain

$$\begin{aligned}\sum_{i=1}^{M} m_i &\le \frac{4}{\zeta} K + 5m + \frac{2\sqrt{K}}{\zeta}(2\sqrt{2}\sqrt{K} + 4\zeta)\\ &= (4 + 4\sqrt{2})\frac{K}{\zeta} + 8\sqrt{K} + 5m\\ &\le (4 + 4\sqrt{2})\frac{K}{\zeta} + (5 + 4\sqrt{2})m\end{aligned}$$

because $K \le m^2/2$. □

Finally, we analyze the running time of the algorithm.

Lemma 4.14. *It requires $O((m + K/\zeta) \log m)$ time to compute all subquadrilaterals $\mathscr{Q}_1, \ldots, \mathscr{Q}_M$, to determine the lines passing through each $\mathscr{Q}_i$, and to count the number of intersection points in each $\mathscr{Q}_i$.*

Proof. It follows from Lemma 4.5 that $\mathbf{Q}^r$, $\mathbf{Q}^g$, and $\mathbf{Q}^b$ can be computed in time $O((m_r + m_b + m_g) \log m) = O(m \log m)$. Once we have computed all pseudoedges, the set $\mathbf{F}$ of cells, obtained by overlaying these pseudoedges, can be computed, for example, by sweeping a vertical line through the quadrilateral $\mathscr{Q}$. The time spent in the sweep is $O((|\mathscr{S}| + |\mathbf{F}|) \log \mathscr{S}) = O((m/\zeta + K/\zeta^2) \log m)$, because it encounters at most $O(|\mathbf{F}|)$ intersection points. Since step 4 simply scans all boundary cells and splits those having more than four edges, it can be easily done in $O(|\mathscr{S}|)$ time.

As for the final step, let m_α (resp. K_α) be the number of lines meeting (resp. intersection points contained in) the αth cell produced by the first four steps. The lines passing through each cell $\mathscr{F}_\alpha$ can be computed in time $O(m \log m + \sum_\alpha m_\alpha \log m) = O((m + K/\zeta) \log m)$ (see Lemma 4.13) by tracing every line through the cells that it intersects, and spending $O(\log m)$ time at each such cell. It follows from Lemma 3.1 that the quantities K_α, for all cells $\mathscr{F}_\alpha$, can be calculated in time $O((m + K/\zeta) \log m)$. Moreover, step 5 of the algorithm partitions the αth cell into at most $\lceil K_\alpha/\zeta\sqrt{K} \rceil$ convex quadrilaterals, which is done by applying the algorithm of Section 3 at most $\lceil K_\alpha/\zeta\sqrt{K} \rceil - 1$ times, each application taking $O(m_\alpha \log m_\alpha)$ time (see Theorem 3.3). Therefore, the total time spent in the final step is bounded by

$$
\begin{aligned}
O\left(\sum_\alpha \left(\left\lceil \frac{K_\alpha}{\zeta\sqrt{K}} \right\rceil - 1\right) m_\alpha \log m_\alpha\right)
&= O\left(\frac{1}{\zeta\sqrt{K}} \sum_\alpha K_\alpha \cdot m_\alpha \log m_\alpha\right) \\
&= O\left(\frac{\sqrt{K}+\zeta}{\zeta\sqrt{K}} \log m \left(\sum_\alpha K_\alpha\right)\right) \qquad (\text{because } m_\alpha \le 2\sqrt{2}\sqrt{K} + 4\zeta) \\
&= O\left(\left(\frac{K}{\zeta} + \sqrt{K}\right) \log m\right) \\
&= O\left(\left(\frac{K}{\zeta} + m\right) \log m\right) \qquad \left(\text{because } K \le \frac{m^2}{2}\right).
\end{aligned}
$$

Thus, the overall running time of our algorithm is $O((m + K/\zeta) \log m)$. □

Hence, we can conclude that

Theorem 4.15. *We can partition the convex quadrilateral $\mathcal{Q}$, in time $O((m + K/\zeta) \log m)$, into $M \le 3m/\zeta + 2K/\zeta^2$ convex subquadrilaterals $\mathcal{Q}_1, \dots, \mathcal{Q}_M$, with the property that each $\mathcal{Q}_i$ is crossed by m_i lines and contains K_i intersection points, so that the following conditions are satisfied*:

(i) $\sum_{i=1}^{M} m_i \le A_1 m + A_2(K/\zeta)$; $\max m_i \le A_3\sqrt{K} + A_4\zeta$;

(ii) $\sum_{i=1}^{M} K_i = K$; $\max K_i \le \zeta\sqrt{K}$,

where $A_1 = (5 + 4\sqrt{2})$, $A_2 = (4 + 4\sqrt{2})$, $A_3 = 2\sqrt{2}$, and $A_4 = 6$. We can also compute the set of lines intersecting the interior of each $\mathcal{Q}_i$, and the values of K_i.

Remark 4.16. If some other simpler sorting network of $O(\log^2 n)$ depth is used in the final step of our algorithm, then the running time will be $O((m + K/\zeta) \log^2 m)$ instead of $O(m + K/\zeta) \log m)$.

5. Partitioning the Plane into Quadrilaterals

In this section we obtain the main result of this paper. Let $\mathcal{L} = \{\ell_1 \ell_2, \dots, \ell_n\}$ be a set of n lines in the plane, and let **R** be an "enclosing rectangle" of $\mathcal{L}$, i.e., one that contains all $\binom{n}{2}$ intersection points of $\mathcal{L}$. For a given integer $r > 0$, we want to partition **R** into $O(r^2 \log^\beta r)$ convex quadrilaterals, for some constant $\beta > 0$ to be determined later, so that each of them is crossed by at most n/r lines.

The idea is to use the algorithm of the previous section recursively. Fix $\zeta = n/\gamma r$, where γ is a constant to be chosen later (for simplicity let us assume that n is a multiple of γr). At each step of the algorithm we have a convex quadrilateral $\mathcal{Q}$, which meets m of the given lines and contains K of their intersection points. Initially $\mathcal{Q} = \mathbf{R}$, $m = n$, and $K = \binom{n}{2}$ (assuming that the lines are in general position). At each recursive step, the algorithm proceeds as follows: if $m \le n/r$, there is nothing to do, so we stop; otherwise partition $\mathcal{Q}$ into M convex quadrilaterals using the algorithm of the previous section (for the initial fixed value of ζ).

Let m_i denote the number of lines meeting the convex quadrilateral $\mathcal{Q}_i$ and let K_i denote the number of intersection points of $\mathcal{L}$ contained in $\mathcal{Q}_i$. Let $\mathcal{C}(m, K)$ denote the maximum number of cells into which such a $\mathcal{Q}$ will be partitioned by all subsequent recursive applications of the algorithm, and let $\mathcal{T}(m, K)$ denote the maximum time required for such a partitioning. It follows from Theorem 4.15 that

$$\mathcal{C}(m, K) \le \begin{cases} \sum_{i=1}^{M} \mathcal{C}(m_i, K_i) & \text{if } m > \frac{n}{r}, \\ 1 & \text{otherwise}, \end{cases} \tag{5.1}$$

$$\mathcal{T}(m, k) \le \begin{cases} \sum_{i=1}^{M} \mathcal{T}(m_i, K_i) + D\left(m + \frac{K}{\zeta}\right)\log m & \text{if } m > \frac{n}{r}, \\ D & \text{otherwise}, \end{cases} \tag{5.2}$$

where $D > 0$ is some constant and M, m_i, and K_i satisfy the bounds given in Theorem 4.15.

Next we bound the values of $\mathcal{C}(m, K)$ and $\mathcal{T}(m, K)$ using (5.1) and (5.2). In what follows by Log x we mean $\max\{\log x, 1\}$, and similarly, later, by Log Log x we mean $\max\{\log\log x, \log\log\sqrt{6}\}$.

Lemma 5.1. *There exists a constant $E > 0$ such that*

$$\mathcal{C}(m, K) \le E\left(\frac{m}{\zeta} + \frac{K}{\zeta^2}\right)\operatorname{Log}^{\beta}\frac{K}{\zeta^2}, \tag{5.3}$$

where $\beta = \max\{\log A_1, \log(1 + A_2)\} < 3.33$.

Proof. We prove the above inequality by induction on K. If we choose $\gamma \ge 4(\sqrt{3} + 1)$, then for $K \le 6\zeta^2$, after applying the algorithm once, we have

$$\begin{aligned} \max m_i &\le \max\{2\sqrt{2}\sqrt{K} + 4\zeta, 2\sqrt{K} + 6\zeta\} \\ &= \max\{(4\sqrt{3} + 4)\zeta, (2\sqrt{6} + 6)\zeta\} \\ &= 4(\sqrt{3} + 1)\zeta \le n/r \end{aligned}$$

and therefore, the algorithm stops. By Theorem 4.15, this step partitions $\mathcal{Q}$ into at most $3\,(m/\zeta + K/\zeta^2)$ cells. Thus, if we choose $E > 3$, then for $K \le 6\zeta^2$, (5.3) holds trivially.

For $K > 6\zeta^2$, suppose inductively that (5.3) holds for all $K' < K$. Since $K_i \leq \zeta\sqrt{K}$ and $K > 6\zeta^2$, $K_i \leq (\sqrt{K/6})\sqrt{K} = K/\sqrt{6} < K$. Therefore by induction hypothesis, (5.3) holds for all $\mathcal{Q}_i$, so (5.1) implies

$$\mathscr{C}(m, K) \leq \sum_{i=1}^{M} E\left(\frac{m_i}{\zeta} + \frac{K_i}{\zeta^2}\right) \operatorname{Log}^\beta \frac{K_i}{\zeta^2}. \tag{5.4}$$

Since $\forall i$, $K_i \leq \zeta\sqrt{K}$, we have

$$\operatorname{Log} \frac{K_i}{\zeta^2} \leq \operatorname{Log} \frac{\zeta\sqrt{K}}{\zeta^2} = \max\left\{\log\sqrt{\frac{K}{\zeta^2}}, 1\right\}.$$

$K \geq 6\zeta^2$ implies that $\log\sqrt{K/\zeta^2} > 1$, therefore (5.4) can be written as

$$\begin{aligned}
\mathscr{C}(m, K) &\leq \sum_{i=1}^{M} E\left(\frac{m_i}{\zeta} + \frac{K_i}{\zeta^2}\right) \log^\beta \sqrt{\frac{K}{\zeta^2}} \\
&= E\left(\frac{1}{\zeta}\sum_{i=1}^{M} m_i + \frac{1}{\zeta^2}\sum_{i=1}^{M} K_i\right) \cdot \frac{1}{2^\beta} \log^\beta \frac{K}{\zeta^2} \\
&\leq \frac{E}{2^\beta}\left(A_1\frac{m}{\zeta} + (A_2 + 1)\frac{K}{\zeta^2}\right) \log^\beta \frac{K}{\zeta^2} \qquad \text{(by Theorem 4.15).}
\end{aligned}$$

Let $\beta = \max\{\log A_1, \log(1 + A_2)\}$, then

$$\frac{1}{2^\beta}\left(A_1\frac{m}{\zeta} + (A_2 + 1)\frac{K}{\zeta^2}\right) \leq \left(\frac{m}{\zeta} + \frac{K}{\zeta^2}\right).$$

Therefore

$$\begin{aligned}
\mathscr{C}(m, K) &\leq E\left(\frac{m}{\zeta} + \frac{K}{\zeta^2}\right) \log^\beta \frac{K}{\zeta^2} \\
&\leq E\left(\frac{m}{\zeta} + \frac{K}{\zeta^2}\right) \operatorname{Log}^\beta \frac{K}{\zeta^2}.
\end{aligned}$$ □

Remark 5.2. The actual value of β depends on the constants appearing in Theorem 4.15(i). We believe that these constants are not optimal, so β is likely to be smaller than $\log_2(5 + 4\sqrt{2}) = 3.33$.

Next we bound the running time $\mathscr{T}(m, K)$ of the algorithm.

Lemma 5.3. *There exists a constant* $F > 0$ *such that*

$$\mathscr{T}(m, K) \leq F\left(m + \frac{K}{\zeta}\right) \log m \cdot \operatorname{Log}^\beta \frac{K}{\zeta^2} \cdot \operatorname{Log}\operatorname{Log} \frac{K}{\zeta^2}. \tag{5.5}$$

Proof. Again we prove the inequality by induction on K. In Lemma 5.1 we showed that, for $K \le 6\zeta^2$, the algorithm stops after one step of recursion. By Theorem 4.15, this step requires $O((m + K/\zeta)\log m)$ time, therefore if we choose F sufficiently large, then (5.5) holds trivially.

For $K > 6\zeta^2$, suppose inductively that (5.5) holds for all $K' < K$. Since $K_i < K$ (see Lemma 5.1), by induction hypothesis, (5.5) holds for all $\mathscr{Q}_i$, so (5.2) implies

$$\mathscr{T}(m, K) \le \sum_{i=1}^{M} F\left(m_i + \frac{K_i}{\zeta}\right)\log m_i \cdot \mathrm{Log}^\beta \frac{K_i}{\zeta^2} \cdot \mathrm{Log\,Log}\frac{K_i}{\zeta^2} + D\left(m + \frac{K}{\zeta}\right)\log m. \tag{5.6}$$

Note that, for all i, $K_i \le \zeta\sqrt{K}$, therefore

$$\mathrm{Log}^\beta \frac{K_i}{\zeta^2}\,\mathrm{Log\,Log}\frac{K_i}{\zeta^2} \le \max\left\{\log^\beta \sqrt{\frac{K}{\zeta^2}}, 1\right\}\cdot \max\left\{\log\log\sqrt{\frac{K}{\zeta^2}}, \log\log\sqrt{6}\right\}.$$

But $K \ge 6\zeta^2$, therefore it is easily seen that

$$\mathrm{Log}^\beta \sqrt{\frac{K}{\zeta^2}}\,\mathrm{Log\,Log}\sqrt{\frac{K}{\zeta^2}} = \log^\beta \sqrt{\frac{K}{\zeta^2}}\log\log\sqrt{\frac{K}{\zeta^2}}. \tag{5.7}$$

Substituting it in (5.6), we obtain

$$\begin{aligned}
\mathscr{T}(m, K) &\le \sum_{i=1}^{M} F\left(m_i + \frac{K_i}{\zeta}\right)\log m\cdot\log^\beta\sqrt{\frac{K}{\zeta^2}}\cdot\log\log\sqrt{\frac{K}{\zeta^2}} + D\left(m + \frac{K}{\zeta}\right)\log m\\
&\le \left(\sum_{i=1}^{M} m_i + \frac{1}{\zeta}\sum_{i=1}^{M} K_i\right)\frac{F}{2^\beta}\cdot\log^\beta\frac{K}{\zeta}\left(\log\log\frac{K}{\zeta^2} - 1\right)\log m + D\left(m + \frac{K}{\zeta}\right)\log m\\
&\le \left(A_1 m + (A_2 + 1)\frac{K}{\zeta}\right)\frac{F}{2^\beta}\cdot\log^\beta\frac{K}{\zeta^2}\left(\log\log\frac{K}{\zeta^2} - 1\right)\log m\\
&\quad + D\left(m + \frac{K}{\zeta}\right)\log m \qquad \text{(by Theorem 4.15)}\\
&\le \left(m + \frac{K}{\zeta}\right)\left[F\log^\beta\frac{K}{\zeta^2}\cdot\log\log\frac{K}{\zeta^2} + \left(D - F\log^\beta\frac{K}{\zeta^2}\right)\right]\log m
\end{aligned}$$

because $\beta = \max\{\log A_1, \log(A_2 + 1)\}$. Since $K > 6\zeta^2$, we have $\log^\beta(K/\zeta^2) > 2^\beta$, and therefore, by choosing $F = D/2^\beta$, we obtain

$$\begin{aligned}
\mathscr{T}(m, K) &\le F\left(m + \frac{K}{\zeta}\right)\log m\cdot\log^\beta\frac{K}{\zeta^2}\cdot\log\log\frac{K}{\zeta^2}\\
&\le F\left(m + \frac{K}{\zeta}\right)\log m\cdot\mathrm{Log}^\beta\frac{K}{\zeta^2}\cdot\mathrm{Log\,Log}\frac{K}{\zeta^2}. \qquad \square
\end{aligned}$$

These lemmas imply that

Theorem 5.4. *Given a set $\mathscr{L}$ of n lines in the plane and a parameter $1 < r < n$, we can partition an enclosing rectangle* **R** *into $O(r^2 \log^\beta r)$ convex quadrilaterals in time $O(nr \log n \log^\beta r \log\log r)$, so that each quadrilateral meets at most n/r lines of $\mathscr{L}$.*

Proof. Choose $\zeta = n/\gamma r$, where $\gamma = 4(\sqrt{3} + 1)$. At the top level of the recursion $m = n$ and

$$K = \binom{n}{2} < \frac{n^2}{2}.$$

Therefore

$$\frac{K}{\zeta^2} < \frac{n^2/2}{n^2/\gamma^2 r^2} = \frac{\gamma^2 r^2}{2}$$

and $m/\zeta = n/(n/\gamma r) = \gamma r$. Substituting these values in (5.3) we obtain

$$\begin{aligned}\mathscr{C}\left(n, \frac{n^2}{2}\right) &\le E\left(\gamma r + \frac{\gamma^2 r^2}{2}\right) \operatorname{Log}^\beta \frac{\gamma^2 r^2}{2}\\ &= O(r^2 \log^\beta r)\end{aligned}$$

and by substituting $K/\zeta = (n^2/2)/(n/\gamma r) = (\gamma r/2)n$ in (5.5) we get

$$\begin{aligned}\mathscr{T}(n, n^2) &\le F\left(n + \frac{\gamma r}{2} n\right) \log n \cdot \operatorname{Log}^\beta \frac{\gamma^2 r^2}{2} \operatorname{Log}\operatorname{Log} \frac{\gamma^2 r^2}{2}\\ &= O(nr \log n \log^\beta r \log\log r).\end{aligned}$$ □

Remark 5.5. (i) As a side product our algorithm outputs, for each cell, the lines intersecting its interior.

(ii) To make the notation easier to follow, we henceforth replace the term $\log^\beta r \cdot \log\log r$ by $\log^\omega r$ for some fixed constant ω slightly larger than β. Since we do not know what the best value for β is, this convention involves no real loss of information.

6. Constructing Approximate Levels

In this section we describe the second phase of our algorithm that reduces the number of triangles from $O(r^2 \log^\beta r)$ to $O(r^2)$, while maintaining the property that each triangle meets $O(n/r)$ lines of $\mathscr{L}$. As mentioned in the introduction, this second phase is not required in most of the applications. As an intermediate step, the algorithm constructs an $(n/2r)$-approximate leveling of $\mathscr{A}(\mathscr{L})$ (as defined in Section 2), with $O(r^2 \log^\beta r)$ edges in total, from the partition obtained in the first phase of the algorithm. Once an $(n/2r)$-approximate leveling has been constructed, we proceed in the same way as Matoušek [Ma]. We first describe how to obtain an $(n/2r)$-approximate leveling.

Let $\bar{\mathscr{Q}}$ denote the planar map induced by the preceding partition of the enclosing rectangle $\mathbf{R}$. We assume that all lines in $\mathscr{L}$ intersect $\partial\mathbf{R}$ at its vertical edges. Let $A = \{a_1, a_2, \ldots, a_n\}$ (resp. $B = \{b_1, b_2, \ldots, b_n\}$) denote the intersection points of the lines in $\mathscr{L}$ and the left (resp. right) vertical edge of $\partial\mathbf{R}$, sorted in decreasing order of their y-coordinates. For $1 \le i \le r$, add $a_{in/r}$ and $b_{in/r}$ to the set of vertices of $\bar{\mathscr{Q}}$. We triangulate all faces of $\bar{\mathscr{Q}}$, in time $O(r^2 \log^\beta r)$. The triangulated map $\mathscr{G}$ also has $O(r^2 \log^\beta r)$ edges. The following observations enable us to compute approximate levels efficiently.

Lemma 6.1. *Let $\mathscr{K}_a$ (resp. $\mathscr{K}_b$) denote the $k - n/2r$ (resp. $k + n/2r$) level of $\mathscr{A}(\mathscr{L})$. Then the interior of a triangle in $\bar{\mathscr{Q}}$ does not intersect both $\mathscr{K}_a$ and $\mathscr{K}_b$.*

Proof. Assume to the contrary that there is such a triangle $\Delta = v_i v_j v_k$ that crosses $\mathscr{K}_a$ and $\mathscr{K}_b$, then Δ contains a point p of the level $k - n/2r - 1$ and another point q of level $k + n/2r$ (see Fig. 14). Obviously, the segment $\overline{p_a p_b}$ lies entirely in Δ, but by Lemma 2.1 it intersects at least $n/r + 1$ lines of $\mathscr{L}$, which contradicts the property that no triangle in $\mathscr{G}$ intersects more than n/r lines of $\mathscr{L}$. □

Lemma 6.2. *Let $\mathscr{K}_a$ (resp. $\mathscr{K}_b$) denote the $k - n/2r$ (resp. $k + n/2r$) level of $\mathscr{A}(\mathscr{L})$. Let v_0 be a vertex of $\mathscr{G}$ on the left vertical edge of $\mathbf{R}$, whose level is between $k - n/2r$ and $k + n/2r$. There exists a path Π in $\mathscr{G}$ from v_0 to the right vertical edge of $\mathbf{R}$ which lies between $\mathscr{K}_a$ and $\mathscr{K}_b$. Moreover Π can be converted into a monotone path without increasing its number of edges, such that the new path also lies between $\mathscr{K}_a$ and $\mathscr{K}_b$.*

Proof. Let $\Delta = \Delta_1, \Delta_2, \ldots, \Delta_t$ denote the sequence of triangles visited if we follow $\mathscr{K}_b$ from left to right (if a portion of $\mathscr{K}_b$ lies on an edge of a triangle, then we pick up the triangle lying below that edge). If a triangle appears more than once in Δ, then retain only its first occurrence. It is easily seen that Δ forms a connected region from the left vertical edge to the right vertical edge of $\partial\mathbf{R}$, and its boundary is formed by the edges of $\mathscr{G}$ (see Fig. 15); we call it a *corridor*. Let Π denote the top portion of $\partial\Delta$. Π is a connected polygonal path from the left vertical edge of $\partial\mathbf{R}$ to its right vertical edge, formed by the edges of $\mathscr{G}$. Obviously Π does not intersect $\mathscr{K}_b$.

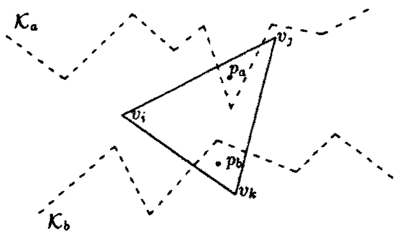

Fig. 14. A triangle intersecting both $\mathscr{K}_a$ and $\mathscr{K}_b$.

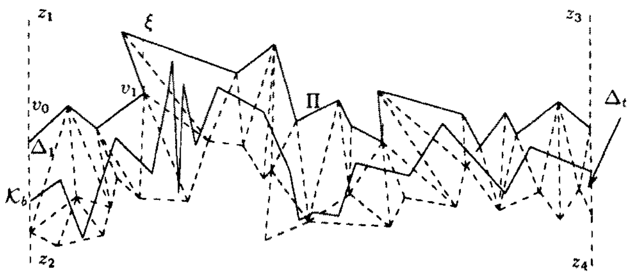

Fig. 15. Level $\mathscr{K}_b$ contained in the corridor Δ; Π is the upper boundary of Δ.

Since each triangle of Δ intersects $\mathscr{K}_b$, Π cannot intersect $\mathscr{K}_a$ (see Lemma 6.1), thus Π lies between $\mathscr{K}_a$ and $\mathscr{K}_b$.

To prove the second part, let Γ denote any polygonal path lying between $\mathscr{K}_a$ and $\mathscr{K}_b$. Let v_1 be the first vertex on Γ at which Γ turns backward. Let ξ denote the first point on Γ after v_1 (along Γ), which has the same x-coordinate as v_1. Since v_1 lies above $\mathscr{K}_b$, and $\mathscr{K}_b$ is a x-monotone path, the vertical segment $\overline{v_1\xi}$ does not cross $\mathscr{K}_b$. The same argument implies that $\overline{v_1\xi}$ does not intersect $\mathscr{K}_a$, and therefore the new path also lies between $\mathscr{K}_a$ and $\mathscr{K}_b$. Whenever we add a new edge, we remove at least one edge of Γ, which proves that shortcutting Γ does not increase its number of edges. □

Since the path Π lies between the $k - n/2r$ and $k + n/2r$ levels of $\mathscr{A}(\mathscr{L})$, it is an $(n/2r)$-approximate k-level of $\mathscr{A}(\mathscr{L})$. Therefore $\mathscr{G}$ contains an $(n/2r)$-approximate (in/r)-level, for each $i \le r$. Let $\mathscr{G}_i$ denote the subgraph of $\mathscr{G}$ consisting of all edges that fully lie between the levels $in/r - n/2r$ and $in/r + n/2r$. To obtain $\mathscr{G}_i$, we need to compute the level of each vertex of $\mathscr{G}$ and the highest and the lowest levels crossed by each edge of $\mathscr{G}$. Let $\psi(v)$ denote the level of a vertex $v \in \mathscr{G}$. Let $\overline{v_iv_j}$ denote an edge of $\mathscr{G}$ with $x(v_i) < x(v_j)$. If there are h lines of $\mathscr{L}$ intersecting $\overline{v_iv_j}$ and k of these lines lie above v_i, then

$$\psi(v_j) = \psi(v_i) - k + (h - k) = \psi(v_i) - 2k + h. \tag{6.1}$$

For each vertex of $\mathscr{G}$ lying on the left vertical edge of $\partial\mathbf{R}$, we can compute its level by counting the number of lines of $\mathscr{L}$ lying above it. The partitioning algorithm produces the subset of lines of $\mathscr{L}$ that crosses each edge of $\mathscr{G}$, so it is trivial to count how many of them lie above the left endpoint of e. The levels of all vertices of $\mathscr{G}$ are now easy to determine by propagating levels from left to right along the edges of $\mathscr{G}$, using (6.1). As for determining the levels crossed by an edge e of $\mathscr{G}$, we sort the lines intersecting e along the edge. Once we know the level of the endpoints of e, we can easily compute the levels crossed by e. Next we partition $\mathscr{G}$ into $\mathscr{G}_1, \ldots, \mathscr{G}_r$, and find a path from $a_{in/r}$ to $b_{in/r}$ in $\mathscr{G}_i$ using a depth-first search from $a_{in/r}$, for $i \le r$. Finally, if any of the resulting paths is not x-monotone, we make it monotone by shortcutting all edges that turn backward. Since the edges in $\mathscr{G}_i$, for $i \le r$, are pairwise disjoint, there are at most $O(r^2 \log^\beta r)$ edges in the resulting $(n/2r)$-approximate leveling of $\mathscr{A}(\mathscr{L})$.

The correctness of the algorithm follows immediately from the above discussion, so we only have to analyze the running time of the algorithm.

Lemma 6.3. *Given a set $\mathscr{L}$ of n lines, we can construct, in time $O(nr \log n \log^{\omega} r)$, an $(n/2r)$-approximate leveling of $\mathscr{A}(\mathscr{L})$ having only $O(r^2 \log^{\beta} r)$ edges in total.*

Proof. By Theorem 5.4, the planar graph $\mathscr{G}$ can be constructed in time $O(nr \log n \cdot \log^{\omega} r)$ and it has only $O(r^2 \log^{\beta} r)$ triangles. It follows from the above discussion that it takes $O(nr \log n \log^{\omega} r)$ time to compute the level of each vertex and the levels crossed by each edge of $\mathscr{G}$. Therefore, we can obtain $\mathscr{G}_i$, for $i \leq r$, in $O(nr \log n \log^{\omega} r)$ time. The depth-first search takes only $O(r^2 \log^{\beta} r)$ time, and it takes the same amount of time to convert the computed paths into monotone paths. Hence, the lemma follows. □

Remark 6.4. Matoušek [Ma] also constructs an approximate leveling as an intermediate step, but in a direct and much simpler (though inefficient) manner. His algorithm works roughly as follows.

Partition $\mathbf{R}$ into r^2 vertical slabs each containing at most $O(n/r^2)$ intersection points of $\mathscr{L}$. Let $V_0, \ldots, V_{r^2}$ denote the vertical edges of these slabs, and let $a_{i,1}, \ldots, a_{i,n}$ denote the intersection points of V_i and $\mathscr{L}$ sorted in decreasing order of their y-coordinates. For each $j \leq r/2$ and $0 \leq i < r^2$ connect $a_{i,2jn/r}$ to $a_{i+1,2jn/r}$. These polygonal paths are shown to form an (n/r)-approximate leveling of $\mathscr{A}(\mathscr{L})$.

The problem with this approach is that the approximate leveling has $O(r^3)$ edges in total, and the time needed to obtain it is $O(nr^2 \log^2 r)$, which is substantially dominated by the partitioning of $\mathbf{R}$ into vertical slabs. Partitioning $\mathbf{R}$ into r^2 slabs is done to ensure that no segment $\overline{a_{i,2jn/r}a_{i+1,2jn/r}}$ crosses too many levels. Using our improved partitioning technique we are able to obtain an $(n/2r)$-approximate partitioning that has almost an order of magnitude fewer edges (in terms of r), and the running time of our algorithm is also about an order of magnitude faster. If r is small, e.g., $O(1)$, then Matoušek's algorithm is better (it runs in optimal linear time), but for large values of r it becomes very inefficient. As we will see in [A], in most of the applications it is desirable to use a large value of r.

After computing an $(n/2r)$-approximate leveling of $\mathscr{A}(\mathscr{L})$, we apply the same technique of Matoušek to partition $\mathbf{R}$ into $O(r^2)$ triangles. Let $K_1, K_2, \ldots, K_r$ denote the set of $(n/2r)$-approximate levels of $\mathscr{A}(\mathscr{L})$. Using Lemmas 2.2 and 2.3, Matoušek proved that

Lemma 6.5 [Ma]. *There exists a polygonal path Π_i between K_{3i-1} and K_{3i+1} such that $\sum_{i=1}^{r/3} |\Pi_i| = O(r^2)$.*

Note that K_{3i-1} lies between the levels $3i(n/r) - \frac{3}{2}\cdot(n/r)$ and $3i(n/r) + \frac{1}{2}\cdot(n/r)$, and K_{3i+1} lies between the levels $3i(n/r) - \frac{1}{2}\cdot(n/r)$ and $3i(n/r) + \frac{3}{2}\cdot(n/r)$. Therefore Π_i lies between the levels $3i(n/r) \mp \frac{3}{2}\cdot(n/r)$, which shows that $\Pi_1, \ldots, \Pi_{r/3}$ forms an $(3n/2r)$-approximate leveling of $\mathscr{A}(\mathscr{L})$. By applying Suri's algorithm [S] of

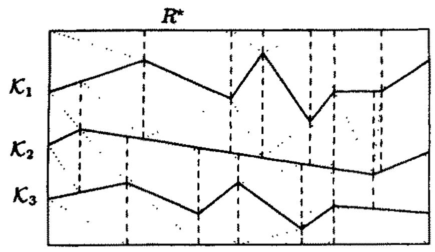

Fig. 16. Triangulation of $\mathscr{K}^*$.

computing a minimum link path in the simply connected region, lying between K_{3i-1} and K_{3i+1}, we obtain a path Π'_i, such that $|\Pi'_i| \le |\Pi_i|$. Since Π'_i lies between K_{3i-1} and K_{3i+1}, it is also a $(3n/2r)$-approximate $(3in/r)$-level. Hence, we have

Corollary 6.6. *Given a set $\mathscr{L}$ of n lines and a parameter $1 < r < n$, we can compute a set of $(3n/2r)$-approximate levels of $\mathscr{A}(\mathscr{L})$, $\mathscr{K} = \{\Pi'_i, \ldots, \Pi'_{r/3}\}$ in time $O(nr \log n \log^{\omega} r)$ with the property that $\sum_{i=1}^{r/3} |\Pi'_i| = O(r^2)$.*

Matoušek has also proved that

Lemma 6.7 [Ma]. *There are at most $O(nr)$ intersections between $\mathscr{K}$ and $\mathscr{L}$.*

In view of Lemma 6.7, $\mathscr{K}$ can be decomposed into $O(r^2)$ edges, each intersecting at most $O(n/r)$ lines of $\mathscr{L}$. Next, construct the vertical decomposition $\mathscr{K}^*$ of $\mathscr{K}$ by drawing a vertical line from every vertex of $\mathscr{K}$ in both directions until it meets an edge of $\mathscr{K}$ or **R** (see Fig. 16). Since every vertical edge added to $\mathscr{K} \cup \mathbf{R}$ lies within $3n/r$ levels, it intersects at most $3n/r$ lines of $\mathscr{L}$. Therefore, every trapezoid of $\mathscr{K}^*$ intersects at most $O(n/r)$ lines of $\mathscr{L}$. Finally, partition each trapezoidal cell of $\mathscr{K}^*$ into two triangles. Hence, we can obtain the main and final result of the paper.

Theorem 6.8. *Given a set $\mathscr{L}$ of n lines in the plane, and a parameter $1 < r < n$, we can decompose the enclosing rectangle* **R** *into $O(r^2)$ triangles in time $O(nr \log n \log^{\omega} r)$, for some constant $\omega < 3.33$, so that no triangle meets more than $O(n/r)$ lines of $\mathscr{L}$.*

Remark 6.9. Once **R** has been partitioned into $O(r^2)$ triangles, we can easily compute, for each triangle, the set of lines passing through its interior by spending $O(nr)$ additional time.

7. Coping with Degenerate Cases

In this section we show how to modify our partitioning algorithm so that it also works in degenerate cases, when more than two lines of $\mathscr{L}$ are concurrent, or more than one intersection point of lines in $\mathscr{L}$ lie on the same vertical line. To facilitate these modifications, we first need to redefine some of the terminology introduced in Sections 2 and 4.

(i) If $t \geq 2$ lines of $\mathscr{L}$ pass through a point p, then we consider p as $\binom{t}{2}$ intersection points. Therefore, although the number of distinct intersection points in $\mathscr{L}$ can be less than $\binom{n}{2}$, the sum of their "weights," as just defined, is still $\binom{n}{2}$.

(ii) The level of a vertex of $\mathscr{A}(\mathscr{H})$ can no longer be uniquely defined, so we redefine the k-level of $\mathscr{A}(\mathscr{H})$ to be the polygonal path formed by the closure of the (open) edges of $\mathscr{A}(\mathscr{H})$ whose level is k. Let $p_0, \ldots, p_m$ be the vertices of a k-level of $\mathscr{A}(\mathscr{H})$. Let w_i denote the weight of p_i, and, for any $\delta < m$, let p_{i_j} be the vertex of a k-level such that $\sum_{t=0}^{i_j-1} w_t < j\delta \leq \sum_{t=0}^{i_j} w_t$. We now define the *$\delta$-simplified k-level* to be the polygonal path connecting p_0 to p_{i_1}, p_{i_1} to $p_{i_2}, \ldots, p_{i_s}$ to p_m, for $s = \lfloor(\sum_t w_t)/\delta\rfloor$, concatenated with the left and right rays of the k-level incident to p_0 and p_m, respectively.

(iii) For a convex quadrilateral $\mathscr{Q}$, we use $\mathscr{L}$ to denote the set of lines passing through the *interior* of $\mathscr{Q}$, and we let K denote the total weight of the intersection points contained in the interior of $\mathscr{Q}$.

(iv) The set A (resp. B) of left (resp. right) endpoints of the lines in $\mathscr{L}$, defined in Section 2, now becomes a multiset, because many lines can have a common endpoint. If two lines ℓ_i and ℓ_j have a common left (resp. right) endpoint x and ℓ_i lies counterclockwise (resp. clockwise) from ℓ_j, when directed from x into $\mathscr{Q}$, then $\pi(i) < \pi(j)$ (resp. $\sigma(i) > \sigma(j)$). In other words, A and B are ordered in the way they should be, if we shrink $\mathscr{Q}$ slightly.

(v) Finally, the quantity δ_i defined in Section 4 denotes the number of pseudoedges whose relative interior intersects ℓ_i. Similarly we define the quantities δ_i^{xy}.

Next we briefly describe the modifications in our algorithm and in its analysis, required to make them work in degenerate cases as well; we leave it for the reader to fill in the details.

Observe that in Section 3.1 we actually count the number of inversions to compute K_{rr}, K_{br}, and K_{gr}, which gives the total weight of the intersection points (not the number of distinct intersection points) contained in $\mathscr{Q}$, so this part of the algorithm does not require any modification. However, now it is not always possible to find a vertical line that, for a given $k \leq K$, has exactly k intersection points in $\mathscr{Q}$ to its left (see Section 3.2). Instead, we find the rightmost vertical line having $\leq k$ intersections in $\mathscr{Q}$ to its left. If we interpret the order of the lines of $\mathscr{L}$ along a vertical line λ as the order that would result by moving λ slightly to the left, then it is easily checked that the procedure described in Section 3.2, with obvious and trivial modifications, would produce the desired line. We now apply this procedure, in the final step of our general partitioning algorithm (see Section 4.2), to partition a cell $\mathscr{F}_\alpha$ into subcells, each containing $\leq \zeta\sqrt{K}$ intersection point in its interior.

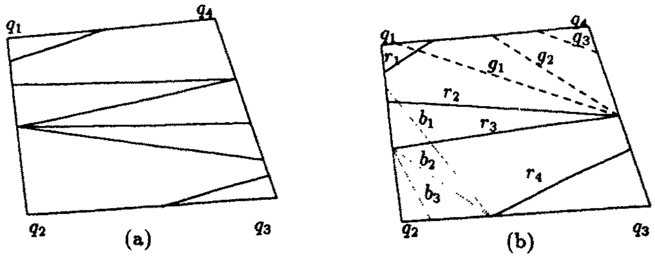

Fig. 17. Cells in degenerate cases: (a) special case; (b) general case.

The algorithm of Section 4.1 remains the same. Since many elements in A or B can have the same value, two or more pseudoedges can have a common endpoint (see Fig. 17(a)) or even completely overlap; however, we regard overlapping pseudoedges as a single edge. It can be checked that our new conventions regarding A and B imply that a line $\ell_i \in \mathscr{L}$ intersects at least $|\pi(i) - \sigma(i)|$ lines inside $\mathscr{Q}$, that is, Lemma 4.1 still holds, which in conjunction with the new definitions of K and δ implies that Lemma 4.2 also continues to hold. Lemma 4.4 is trickier to adjust. We redefine μ_i as the number of lines passing through $\mathscr{Q}_i$ and not having an endpoint whose rank in A or in B is between $(i-1)\zeta + 1$ and $i\zeta$. With some care, the proof can be modified to yield the same bound on μ_i.

Our general algorithm described in Section 4.2 also does not change except that in the final step we now use the modified algorithm mentioned above to partition the cells $\mathscr{F}_\alpha$ into $\lceil K_\alpha/\zeta\sqrt{K}\rceil$ cells, each containing at most $\zeta\sqrt{K}$ intersection points in its interior. Note that now we can have some triangular cells that do not have an edge lying along $\partial\mathscr{Q}$ but only have a vertex lying on $\partial\mathscr{Q}$; these cells are also considered as boundary cells. We may even have some zero-area cells, caused by overlapping pseudoedges, but we discard these cells. In view of our convention regarding the weights of intersection points and the order of the lines along $\partial\mathscr{Q}$, it is easy to see that the charging scheme of Lemma 4.7 still works. (More specifically, we charge a red-green intersection to a red-green cell $\mathscr{F}_{ij}$ if the rank of the right endpoint of the red line is between $(j-1)\zeta + 1$ and $j\zeta$ and the rank of the appropriate endpoint of the green line is between $(i-1)\zeta + 1$ and $i\zeta$, and similarly for red-blue intersections.) Moreover, using the same argument as in Section 4.3, we can prove Lemmas 4.10 and 4.11 in this degenerate setting because we are not counting those pseudolines that intersect ℓ at their endpoints. Finally, it can be shown that Lemmas 4.9, 4.12, and 4.13 also hold, because, although a line ℓ_i can meet the boundary of many cells, it meets the interior of exactly $\delta_i + 1$ cells. Lemma 4.14 can also be shown to hold, using our notational convention in degenerate cases.

As for the second phase of our algorithm, Lemmas 6.1 and 6.2 are not affected by degeneracies in $\mathscr{L}$. However, computing the graphs $\mathscr{G}_k$ now becomes slightly more difficult because the level of a vertex v of $\mathscr{G}$ may be undefined (that is, when v is a "heavy-weighted" vertex of $\mathscr{A}(\mathscr{L})$ as well). But for each vertex v of $\mathscr{G}$, we can compute the lines of $\mathscr{L}$ passing through it in time $O(nr \log n \log^{\omega} r)$, because a line passing through a vertex v either lies in the interior of a triangle adjacent to v, or contains one of the edges adjacent to v. Moreover, for each edge incident to v, we can compute, in time $O(nr \log n \log^{\omega} r)$, its level in a sufficiently small neighborhood of v. Therefore, we can still propagate the levels from left to right along the edges of $\mathscr{G}$, and can determine the levels crossed by each edge of $\mathscr{G}$. Thus, we can partition $\mathscr{G}$ into $\mathscr{G}_1, \ldots, \mathscr{G}_r$ and obtain an $(n/2r)$-approximate leveling of $\mathscr{A}(\mathscr{L})$. It can be checked that the technical results in [EW] and [Ma] can be extended to the degenerate case, which in turn implies that Lemmas 2.3 and 6.5 still hold if we follow our new conventions regarding simplified levels and the weights of intersection points. It is also easy to check that the proof of Lemma 6.7 does not require the lines of $\mathscr{L}$ to be in general position.

Hence, we can conclude that Theorem 6.8 holds, with appropriate modifications as discussed above, even if the lines are not in general position.

8. Conclusions

In this paper we presented a deterministic algorithm that, given a set $\mathscr{L}$ of n lines and a parameter $1 < r < n$, partitions the plane into $O(r^2)$ triangles, each of which meets at most $O(n/r)$ lines of $\mathscr{L}$. Although we showed that our algorithm is optimal up to a polylog factor, there are some questions that still remain unanswered:

(i) The time complexity of our algorithm has an extra $\log^{\omega} r$ factor. The value of ω depends on the constants appearing in the bounds of $\max_i m_i$ and $\sum_{i=1}^{M} m_i$ in Theorem 4.15. We believe the values of these constants can be improved by doing a more careful analysis, so a natural question is to determine the best possible values of these constants.

(ii) As mentioned in the introduction, we conjecture that the second phase of our algorithm is redundant, i.e., the first phase itself or some appropriate variant of it produces $O(r^2)$ triangles; so far we have not been able to prove this conjecture.

(iii) One drawback of our algorithm is that its space complexity is $O(nr)$. This is because at every level of recursion we need to determine the lines passing through each cell. Can the space complexity be reduced to $O(n + r^2)$?

(iv) The $\Omega(nr)$ lower bound applies only if we want to report the lines passing through each triangle. No nontrivial lower bound is known if these crossings need not be reported. A challenging open problem is to establish a similar lower bound in this case too.

Besides the above open problems related to our algorithm, there are several other open problems related to the general partitioning problem.

(i) The most challenging open problem is to generalize our algorithm to three dimensions. That is, for a given set H of n planes in $\mathbb{R}^3$ and a parameter $1 < r < n$, how fast can we partition space into $O(r^3)$ tetrahedra so that no tetrahedron meets more than $O(n/r)$ planes of H? The algorithm of [CF] yields such a partitioning in time $O(n^{10})$. Since the number of plane-tetrahedron crossings is $\Theta(nr^2)$ in the worst case, an ideal solution would be an algorithm whose running time is close to $O(nr^2)$.

(ii) Our algorithm relies on the fact that the objects are straight lines, so it is still an open problem whether a similar algorithm exists for curves. Note that the algorithm of Chazelle and Friedman [CF] works even for curves, but their construction is very inefficient. Moreoever, in the case of curves, the techniques of [(CF], as well as the random-sampling technique, both give a slightly weaker bound, namely, they partition the plane into $O(r^2)$ simply shaped regions, each meeting at most $O((n/r) \log r)$ curves (instead of $O(n/r)$).

(iii) Suppose we have a collection of m blue lines and a collection of n red lines. For a given parameter r, how fast can one partition the plane into $O(r^2)$ triangles, so that each triangle meets $O(m/r)$ blue lines and $O(n/r)$ red lines? Note that the random-sampling technique easily yields a similar partitioning, with each triangle meeting $O((m/r) \log r)$ blue lines and $O((n/r) \log r)$ red lines.

(iv) Suppose we have a collection $\mathscr{G}$ of n segments which intersect at K points. The random-sampling technique shows that, for a given parameter r, we can partition the plane into $O(r + Kr^2/n^2)$ triangles so that no triangle meets more than $O((n/r) \log r)$ segments of $\mathscr{G}$. Can our algorithms be extended to yield a deterministic algorithm that produces that many triangles, each meeting $O(n/r)$ segments, and runs in time close to $O(n + Kr/n)$ (the lower bound on triangle-segment crossing in this case is $\Omega(n + Kr/n)$).

Acknowledgments

I would like to thank my adviser Micha Sharir for encouraging me to work on this problem, for several valuable discussions, and for reading earlier versions of this paper which helped a lot in improving its quality. Thanks are also due to two anonymous referees for their useful comments.

References

[A] P. K. Agarwal, Partitioning arrangements of lines, II: Applications, *Discrete and Computational Geometry* **5** (1990), 533-574.

[AS] P. K. Agarwal and M. Sharir, Red-blue intersection detection algorithms with applications to motion planning and collision detection, *SIAM Journal on Computing* **19** (1990), 297-322.

[AKS] M. Ajtai, J. Komlos, and E. Szemerédi, Sorting in $c \log n$ parallel steps, *Proceedings of the 15th Annual ACM Symposium on Theory of Computing*, 1983, pp. 1-9.

[B] K. E. Batcher, Sorting networks and their applications, *Proceedings of the AFIPS Spring Joint Summer Computer Conference*, vol. 32 (1968), pp. 307-314.

[CF] B. Chazelle and J. Friedman, A deterministic view of random sampling and its use in geometry, *Proceedings of the 29th Annual IEEE Symposium on Foundations of Computer Science*, 1988, pp. 539-549.

[Cl1] K. Clarkson, A probabilistic algorithm for the post office problem, *Proceedings of the 17th Annual ACM Symposium on Theory of Computing*, 1985, pp. 75-84.

[Cl2] K. Clarkson, New applications of random sampling in computational geometry, *Discrete and Computational Geometry* **2** (1987), 195-222.

[Cl3] K. Clarkson, Applications of random sampling in computational geometry, II, *Proceedings of the 4th Annual Symposium on Computational Geometry*, 1988, pp. 1-11.

[CS] K. Clarkson and P. Shor, Algorithms for diametric pairs and convex hulls that are optimal, randomized and incremental, *Proceedings of the 4th Annual Symposium on Computational Geometry*, 1988, pp. 12-17.

[CTV] K. Clarkson, R. E. Tarjan, and C. J. Van Wyk, A fast Las Vegas algorithm for triangulating a simple polygon, *Discrete and Computational Geometry* **4** (1989), 423-432.

[Co] R. Cole, Slowing down sorting networks to obtain faster sorting algorithms, *Journal of the Association for Computing Machinery* **31** (1984), 200-208.

[CSSS] R. Cole, J. Salowe, W. Steiger, and E. Szemerédi, An optimal-time algorithm for slope selection, *SIAM Journal on Computing* **16** (1989), 792-810.

[CSY] R. Cole, M. Sharir, and C. K. Yap, On *k*-hulls and related problems, *SIAM Journal on Computing* **16** (1987), 61-77.

[E] H. Edelsbrunner, *Algorithms in Combinatorial Geometry*, Springer-Verlag, Heidelberg, 1987.

[EGH*] H. Edelsbrunner, L. Guibas, J. Hershberger, R. Seidel, M. Sharir, J. Snoeyink, and E. Welzl, Implicitly representing arrangements of lines or segments, *Discrete and Computational Geometry* **4** (1989), 433-466.

[EGS] H. Edelsbrunner, L. Guibas, and M. Sharir, The complexity and construction of many faces in arrangements of lines or of segments, *Discrete and Computational Geometry* **5** (1990), 161-196.

[EW] H. Edelsbrunner and E. Welzl, Constructing belts in two-dimensional arrangements with applications, *SIAM Journal on Computing* **15** (1986), 271-284.

[GOS] L. Guibas, M. Overmars, and M. Sharir, Ray shooting, implicit point location, and related queries in arrangements of segments, Technical Report 433, Dept. Computer Science, New York University, March 1989.

[HW] D. Haussler and E. Welzl, ε-nets and simplex range queries, *Discrete and Computational Geometry* **2** (1987), 127-151.

[Ma] J. Matoušek, Construction of ε-nets, *Discrete and Computational Geometry*, this issue, 427-448.

[Me] N. Megiddo, Applying parallel computation algorithms in design of serial algorithms, *Journal of the Association of Computing Machinery* **30** (1983), 852-865.

[RS1] J. Reif and S. Sen, Optimal randomized parallel algorithms for computational geometry, *Proceedings of the 16th International Conference on Parallel Processing*, 1987, pp. 270-277.

[RS2] J. Reif and S. Sen, Polling: A new randomized sampling technique for computational geometry, *Proceedings of the 21st Annual ACM Symposium on Theory of Computing*, 1989, pp. 394-404.

[S] S. Suri, A linear algorithm for minimum link paths inside a simple polygon, *Computer Vision, Graphics and Image Processing* **35** (1986), 99-110.

[We] E. Welzl, More on *k*-sets of finite sets in the plane, *Discrete and Computational Geometry* **1** (1986), 83-94.

[Wo] G. Woeginger, Epsilon-nets for half planes, Technical Report B-88-02, Dept. of Mathematics, Free University, Berlin, March 1988.

Received May 20, 1989, and in revised form January 5, 1990.