11. Constructing Ham-Sandwich Cuts in the Plane

Lecture on Wednesday 4th November, 2009 by Michael Hoffmann <hoffmann@inf.ethz.ch>

In continuation of the necklace splitting / ham-sandwich cut problem... However, knowing about the existence of such a line certainly is not good enough. It is easy to turn the proof given above into an $O(n^2)$ algorithm to construct a line that simultaneously bisects both sets. But we can do better.

The algorithm outlined below is not only interesting in itself but also because it illustrates one of the fundamental general paradigms for designing optimization algorithms: *prune & search*. The basic idea behind prune & search is to search the space of possible solutions by at each step excluding some part of this space from further consideration. For instance, if at each step a constant fraction of all possible solutions can be discarded and a single step is linear in the number of solutions to be considered, then for the runtime we obtain a recursion of the form

$$T(n) \leq cn + T\left(n\left(1 - \frac{1}{d}\right)\right) < cn\sum_{i=0}^{\infty}\left(\frac{d-1}{d}\right)^i < cn\frac{1}{1-\frac{d-1}{d}} = cdn ,$$

that is, a linear time algorithm overall. A well-known example of prune & search is binary search: every step takes constant time and about half of the possible solutions can be discarded, resulting in logarithmic runtime overall.

Theorem 11.1 *Let* $R, D \subset \mathbb{R}^2$ *be finite sets of points. Then in* $O(n \log n)$ *time one can find a line that bisects both* R *and* D*. That is, in either open halfplane defined by* ℓ *there are no more than* $|R|/2$ *points from* R *and no more than* $|D|/2$ *points from* D*.*

Proof. We will describe a recursive algorithm $\text{find}(L_1, k_1, L_2, k_2, (x_1, x_2))$, for sets L_1, L_2 of lines in $\mathbb{R}^2$, non-negative integers k_1 and k_2, and a real interval (x_1, x_2), to find an intersection between the k_1-level of $\mathcal{A}(L_1)$ and the k_2-level of $\mathcal{A}(L_2)$, under the assumption that both levels intersect an odd number of times in (x_1, x_2) and they do not intersect at either $x = x_1$ or $x = x_2$ (*odd-intersection property*). Note that the odd-intersection property is equivalent to saying that the level that is above the other at $x = x_1$ is below the other at $x = x_2$. In the end, we are interested in $\text{find}(R^*, (|R| + 1)/2, D^*, (|D| + 1)/2, (-\infty, \infty))$. As argued in the proof of Theorem 10.5, for these arguments the odd-intersection property holds.

First let $L = L_1 \cup L_2$ and find a line μ with median slope in L. Denote by $L_<$ and $L_>$ the lines from L with slope less than and greater then μ, respectively. (Without loss of generality no two points in $R \cup D$ have the same x-coordinate and thus no two lines in L have the same slope.) Pair the lines in $L_<$ with those in $L_>$ arbitrarily to obtain an almost perfect matching in the complete bipartite graph on $L_< \cup L_>$. Denote by I the $\lfloor(|L_<|+|L_>|)/2\rfloor$ points of intersection generated by the pairs chosen, and let j be a point from I with median x-coordinate.

Determine the intersection (j, y_1) of the k_1-level of L_1 with the vertical line $x = j$ and the intersection (j, y_2) of the k_2-level of L_2 with the vertical line $x = j$. If both levels

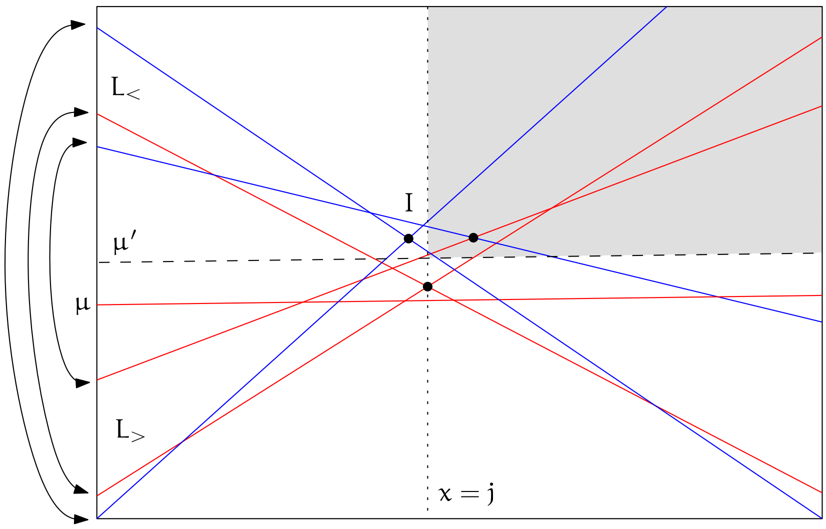

Figure 11.1: *An example with a set L_1 of 4 red lines and a set L_2 of 3 blue lines. Suppose that $k_1 = 2$ and $k_2 = 1$. The interesting quadrant is shaded and the topmost red line (smallest slope) would be discarded.*

intersect at $x = j$, return the intersection and exit. Otherwise, if $j \in (x_1, x_2)$ then exactly one of the intervals (x_1, j) or (j, x_2) has the odd-intersection property, say[1], (x_1, j). In other words, we can from now on restrict our focus to the halfplane $x \leq j$. In a similar way the case $j \notin (x_1, x_2)$ can be handled. For instance, if $j > x_2$ then we know a forteriori that the levels intersect within the halfplane $x \leq j$.

Let $I_>$ denote the set of points from I with x-coordinate greater than j, and let μ' be a line parallel to μ such that about half of the points from $I_>$ are above μ' (and thus the other about half of points from $I_>$ are below μ'). Suppose for the moment that we could tell for one of the halfplanes bounded by μ', say, the one above μ' that the two levels intersect within this halfplane. In other words, we can restrict our focus to the upper left quadrant Q_2 defined by the two lines $x = j$ and μ'. Then by definition of j and μ' about a quarter of the points from I are contained in the opposite, that is, the lower right quadrant Q_4 defined by these two lines. Any point in Q_4 is the point of intersection of two lines from L, one of which has slope larger than μ'. As no line with slope larger than μ' that passes through Q_4 can intersect Q_2, any such line can be discarded from further consideration. In this case, the lines discarded pass completely below the interesting quadrant Q_2. For any line discarded in this way from L_1 or L_2, the parameter k_1 or k_2, respectively, has to be decreased by one. In the symmetric case where the lines discarded pass above the interesting quadrant, the parameters k_1 and k_2 stay the same. In any case, about a 1/8-fraction of all lines in L is discarded.

Note that when discarding lines from L_1 or L_2 or when changing the parameters k_1 or k_2, we cannot guarantee that the odd-intersection property still holds for the original interval (x_1, x_2). Therefore, this interval has to be adjusted at times and appears as an

[1] The other case is completely symmetric and thus will not be discussed here.

additional parameter of the algorithm.

Let us now argue how to find a halfplane defined by μ' that contains an intersection of the two levels. For this it is enough to trace μ' through the arrangement $\mathcal{A}(L)$ while keeping track of the position of the two levels of interest. Initially, at $x = x_1$ we know which level is above the other. At every intersection of one of the two levels with μ', we can check whether the ordering is still consistent with that initial ordering. For instance, if both were above μ' initially and the level that was above the other intersects μ' first, we can deduce that there must be an intersection of the two levels above μ'. As the relative position of the two levels is reversed at $x = x_2$, at some point an inconsistency, that is, the presence of an intersection will be detected and we will be able to tell whether it is above or below μ'. (There could be many more intersections between the two levels, but finding just one intersection is good enough.)

The trace of μ' in $\mathcal{A}(L)$ can be computed by a sweep along μ', which amounts to computing all intersections of μ' with the other lines from L and sorting them by x-coordinate. During the sweep we keep track of the number of lines from L_1 below μ' and the number of lines from L_2 below μ'. At every point of intersection, these counters can be adjusted and any intersection with one of the two levels of interest is detected. Therefore computing the trace takes $O(|L| \log |L|)$ time. This step dominates the whole algorithm, noting that all other operations are based on rank-i element selection, which can be done in linear time.

Altogether, we obtain as a recursion for the runtime $T(n) \leq cn \log n + T(7n/8) = O(n \log n)$. □

You can also think of the two point sets as a discrete distribution of a ham sandwich that is to be cut fairly, that is, in such a way that both parts have the same amount of ham and the same amount of bread. That is where the name "ham sandwich cut" comes from. The theorem also holds in $\mathbb{R}^d$, saying that any d finite point sets (or finite Borel measures, if you want) can simultaneously be bisected by a hyperplane. This implies that the thieves can fairly distribute a necklace consisting of d types of gems using at most d cuts.

The algorithm described above is due to Edelsbrunner and Waupotitsch [1].

Questions

30. *How can one construct an arrangement of lines in* $\mathbb{R}^2$*?* Describe the incremental algorithm and prove that its time complexity is quadratic in the number of lines (incl. statement and proof of the Zone Theorem).

31. *How can one test whether there are three collinear points in a set of* n *given points in* $\mathbb{R}^2$*?* Describe an $O(n^2)$ time algorithm.

32. *How can one compute the minimum area triangle spanned by three out of* n *given points in* $\mathbb{R}^2$*?* Describe an $O(n^2)$ time algorithm.

33. *What is a ham-sandwich cut? Does it always exist? How to compute it?* State and prove the theorem about the existence of a ham-sandwich cut in $\mathbb{R}^2$. Describe an $O(n^2)$ algorithm to compute it and sketch the $O(n \log n)$ algorithm by Edelsbrunner and Waupotitsch.

References

[1] H. Edelsbrunner and R. Waupotitsch, Computing a ham-sandwich cut in two dimensions, *J. Symbolic Comput.* **2** (1986), 171–178.