Chapter 36
Linear time algorithms

By Sariel Har-Peled, December 3, 2018

Version: 0.3

36.1. The lowest point above a set of lines

Let $L$ be a set of $n$ lines in the plane. To simplify the exposition, assume the lines are in general position:

$(A)$ No two lines of $L$ are parallel.
$(B)$ No line of $L$ is vertical or horizontal.
$(C)$ No three lines of $L$ meet in a point.

We are interested in the problem of computing the point with the minimum y coordinate that is above all the lines of $L$. We consider a point on a line to be above it.

For a line $\ell \in L$, and a value $\alpha \in \mathbb{R}$, let $\ell(x)$ be the value of $\ell$ at $\alpha$. Formally, consider the intersection point of $p = \ell \cap (x = \alpha)$ (here, $x = \alpha$ is the vertical line passing through $(\alpha,0)$). Then $\ell(x) = y(p)$.

Let $U_L(\alpha) = \max_{\ell \in L} \ell(\alpha)$ be the upper envelope of $L$. The function $U_L(\cdot)$ is convex, as one can easily verify. The problem asks to compute $y^* = \min_{x \in \mathbb{R}} U_L(x)$. Let $x^*$ be the coordinate such that $y^* = U_L(x^*)$.

Definition 36.1.1. Let $\text{opt}(L) = (x^*, y^*)$ denote the optimal solution – that is, lowest point on $U_L(x)$.

Remark 36.1.2. There are some uninteresting cases of this problem. For example, if all the lines of $L$ have negative slope, then the solution is at $x^* = +\infty$. Similarly, if all the slopes are positive, then the solution is $x^* = -\infty$. We can easily check these cases in linear time. In the following, we assume that at least one line of $L$ has positive slope, and at least one line has a negative slope.

Lemma 36.1.3. Given a value $x$, and a set $L$ of $n$ lines, one can in linear time do the following:

---

This work is licensed under the Creative Commons Attribution-Noncommercial 3.0 License. To view a copy of this license, visit [http://creativecommons.org/licenses/by-nc/3.0/](http://creativecommons.org/licenses/by-nc/3.0/) or send a letter to Creative Commons, 171 Second Street, Suite 300, San Francisco, California, 94105, USA.
Lemma 36.1.7 (Prune). Let \( L \) be a set of \( n \) lines. One can compute, in linear time, either:

(A) A set \( L' \subseteq L \) such that \( \text{opt}(L) = \text{opt}(L') \), and \( |L'| \leq (7/8) |L| \).

Proof: (A) Computing \( \ell(x) \) for \( x \in \mathbb{R} \), takes \( O(1) \) time. Thus computing this value for all the lines of \( L \) takes \( O(n) \) time, and the maximum can be computed in \( O(n) \) time.

(B) For case (I) to happen, there must be two lines that realizes \( U_L(x) \) – one of them has a positive slope, the other has negative slope. This clearly can be checked in linear time.

Otherwise, consider \( U_L(x) \). If there is a single line that realizes the maximum for \( x \), then its slope is the slope of \( U_L(x) \) at \( x \). If this slope is positive than \( x^* < x \). If the slope is negative then \( x < x^* \).

The slightly more challenging case is when two lines realizes the value of \( U_L(x) \). That is \( (x, U_L(x)) \) is an intersection point of two lines of \( L \) (i.e., a vertex) on the upper envelope of the lines of \( L \). Let \( \ell_1, \ell_2 \) be these two lines, and assume that slope(\( \ell_1) < \) slope(\( \ell_2) \).

If slope(\( \ell_2) < 0 \), then both lines have negative slope, and \( x^* > x \). If slope(\( \ell_1) > 0 \), then both lines have positive slope, and \( x^* < x \). If slope(\( \ell_1) < 0 \), and slope(\( \ell_1) > 0 \), then this is case (I), and we are done.

Lemma 36.1.4. Let \( (x, y) \) be the intersection point of two lines \( \ell_1, \ell_2 \in L \), such that slope(\( \ell_1) < \) slope(\( \ell_2) \), and \( x < x^* \). Then \( \text{opt}(L) = \text{opt}(L - \ell_1) \), where \( L - \ell_1 = L \setminus \{ \ell_1 \} \).

Proof: See Figure 36.2. Since \( x < x^* \), it must be that \( U_L(\cdot) \) has a negative slope at \( x \) (and also immediately to its right). In particular, for any \( x \), we have that \( U_L(\alpha) = \ell_2(x) > \ell_1(x) \). That is, the line \( \ell_1(x) \) is “buried” below \( \ell_2 \), and can not touch \( U_L(\cdot) \) to the right of \( x \). In particular, removing \( \ell_1 \) from \( L \) can not change \( U_L(\cdot) \) to the right of \( x \). Furthermore, since \( U_L(\cdot) \) has negative slope immediately after \( x \), it implies that minimum point can not move by the deletion of \( \ell_1 \). Thus implying the claim.

Lemma 36.1.5. Let \( (x, y) \) be the intersection point of two lines \( \ell_1, \ell_2 \in L \), such that slope(\( \ell_1) < \) slope(\( \ell_2) \), and \( x^* < x \). Then \( \text{opt}(L) = \text{opt}(L - \ell_2) \).

Proof: Symmetric argument to the one used in the proof of Lemma 36.1.4.

Observation 36.1.6. The point \( p = \text{opt}(L) \) is a vertex formed by the intersection of two lines of \( L \).

Indeed, since none of the lines of \( L \) are horizontal, if \( p \) was in the middle of a line, then we could move it and improve the value of the solution.

Lemma 36.1.7 (Prune). Given a set \( L \) of \( n \) lines, one can compute, in linear time, either:

(A) A set \( L' \subseteq L \) such that \( \text{opt}(L) = \text{opt}(L') \), and \( |L'| \leq (7/8) |L| \).
(B) A value \( x \) such that \( x^*(L) = x \).

Proof: If \( |L| = n = O(1) \) then one can compute \( \text{opt}(L) \) by brute force. Indeed, compute all the \( \binom{n}{2} \) vertices induced by \( L \), and for each one of them check if they define the optimal solution using the algorithm of Lemma 36.1.3. This takes \( O(1) \) time, as desired.

Otherwise, pair the lines of \( L \) in \( N = \lfloor n/2 \rfloor \) pairs \( \ell_i, \ell'_i \). For each pair, let \( x_i \) be the \( x \)-coordinate of the vertex \( \ell_i \cap \ell'_i \). Compute, in linear time, using median selection, the median value \( z \) of \( x_1, \ldots, x_N \). For the sake of simplicity of exposition assume that \( x_i < z \), for \( i = 1, \ldots, N/2-1 \), and \( x_i > z \), for \( i = N/2+1, \ldots, N \) (otherwise, reorder the lines and the values so that it happens).

Using the algorithm of Lemma 36.1.3 decide which of the following happens:

(I) \( z = x^* \): we found the optimal solution, and we are done.

(II) \( z < x^* \). But then \( x_i < z < x^* \), for \( i = 1, \ldots, N/2-1 \), By Lemma 36.1.4, either \( \ell_i \) or \( \ell'_i \) can be dropped without effecting the optimal solution, and which one can be dropped can be decided in \( O(1) \) time. In particular, let \( L' \) be the set of lines after we drop a line from each such pair. We have that \( \text{opt}(L') = \text{opt}(L) \), and \( |L'| = n - (N/2 - 1) \leq (7/8)n \).

(III) \( z > x^* \). This case is handled symmetrically, using Lemma 36.1.5.

Theorem 36.1.8. Given a set \( L \) of \( n \) lines in the plane, one can compute the lowest point that is above all the lines of \( L \) (i.e., \( \text{opt}(L) \)) in linear time.

Proof: The algorithm repeatedly apply the pruning algorithm of Lemma 36.1.7. Clearly, by the above, this algorithm computes \( \text{opt}(L) \) as desired.

In the \( i \)th iteration of this algorithm, if the set of lines has \( n_i \) lines, then this iteration takes \( O(n_i) \) time. However, \( n_i \leq (7/8)^i n \). In particular, the overall running time of the algorithm is

\[
O\left(\sum_{i=0}^{\infty} (7/8)^i n\right) = O(n).
\]

36.2. Bibliographical notes

The algorithm presented in Section 36.1 is a simplification of the work of Megiddo [Meg84]. Megiddo solved the much harder problem of solving linear programming in constant dimension in linear time. The algorithm presented is essentially the core of his basic algorithm.
Bibliography