# Final Exam Geometric Algorithms, March 19, 2008, 14–17

Read every question carefully, make sure you understand it, and be sure to answer the question. Read the question again after answering it, as a check. Make the easier questions first, and then the harder ones. You may not use any algorithmic result from the book, unless it is explicitly stated that this is allowed in the question. Answer questions in sufficient but not too much detail. You may not use the textbook during the exam.

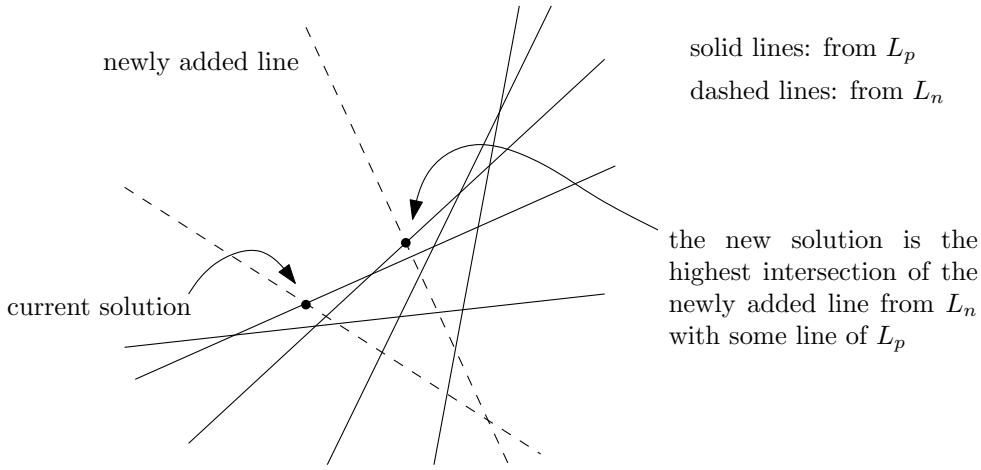
Be sure to put your name on every piece of paper you hand in. Also write down your "collegekaartnummer". Good luck!

1. Let $H$ be a set of half-planes that are all positive, that is, the region above a line. Let $L$ be the set of lines bounding the half-planes of $H$; assume none is vertical or horizontal. The objective is to find the lowest point in the common intersection of the half-planes. This problem can be solved simply by linear programming (as in Chapter 4 of the book), but here we look at a variation.

   Assume we partition $L$ into two subsets $L_p$ and $L_n$, where $L_p$ contains all lines of $L$ with positive slope, and $L_n$ contains all lines of $L$ with negative slope (recall that no line is vertical or horizontal). Take any line of $\ell \in L_p$ and any line of $\ell' \in L_n$ and let their intersection be the current solution.

   Phase 1: We put the remaining lines of $L_p$ in random order and insert them one by one, as usual with randomized incremental construction. If the next line lies below the current solution, then it stays the same. If it does not lie below the current solution, then the new solution is the intersection of the line just added with $\ell'$.

   Phase 2: We put the remaining lines of $L_n$ in random order and insert them one by one. If the next line lies below the current solution, then it stays the same. If it does not lie below the current solution, then the new solution is the intersection of the line just added with some line from $L_p$, namely the one that has the *highest intersection point* on the newly added line. See the figure. We find the new solution by going over all lines of $L_p$ to find this highest intersection point, before continuing with the next line of $L_n$.




**(a.)** (0.5 point) What is the worst case running time of phase 1 of this algorithm? What is the worst case running time of phase 2 of this algorithm? Motivate your answers.

**(b.)** (1 point) What is the expected running time of phase 1 of this algorithm? What is the expected running time of phase 2 of this algorithm? Prove your answer (obviously, using backwards analysis somewhere in the proof).

2. (1 point) A simple polygon $P$ is convex if and only if any line that intersects $P$, without containing a vertex of $P$, intersects exactly two edges of $P$.

   Translate this statement into its dual setting.

3. Consider the sweep line algorithm to compute the Voronoi diagram of a set of $n$ points in the plane.

   **(a)** (0.5 point) What is the maximum number of arcs that can occur on the beach line (an exact answer is requested)?

   **(b)** (0.5 point) Potential circle events can be false alarms. What happens geometrically if a circle event that could have happened in the future, is detected to be a false alarm?

   **(c)** (0.5 point) There are several possible degeneracies that can occur during the algorithm. Describe what happens if a site event occurs vertically under a breakpoint of the beach line. How is this event handled?

4. Consider the problem of orthogonal range searching in a set of $n$ points in the plane.

   **(a)** (1 point) Compare query times and construction times for (two-dimensional) kd-trees and range trees storing $n$ points.

   **(b)** (1 point) Describe the structure of a (two-dimensional) range tree storing $n$ points. Discuss the main tree and the associated structures.

5. (2 points) Consider a set $T$ of $n$ disjoint triangles, a set $G$ of $n$ green points, and a set $R$ of $n$ red points in the plane. For simplicity, we assume that no two points and/or vertices share the same $x$- or $y$-coordinates.

   Give a plane sweep algorithm that, given $T$, $G$, and $R$, reports the number of triangles containing at least one green point but no red points. Your algorithm should run in $O(n \log n)$ time.

   Be sure to discuss the sweep line status, the status structure, the event queue, the different types of events that occur, and how these events should be processed.

6. (1 point) Give a linear-time algorithm to partition a strictly $x$-monotone polygon $P$ into two parts of equal area by means of a vertical segment. (You may assume that you are given a black box routine *AreaInbetween*$(e, e')$, which returns the area of the trapezoid bounded from above and below by $e$ and $e'$, and from the left by the rightmost of the left endpoints of $e$ and $e'$, and from the right by the leftmost of the right endpoints of $e$ and $e'$.)

7. (1 point) Determine the area of the Minkowski sum of an axis-parallel rectangle with side lengths 1 and 2 and an axis parallel square with side length 1.