Lehrstuhl für Informatik I
Effiziente Algorithmen und
wissensbasierte Systeme
Universität Würzburg

Würzburg, January 14, 2015

Prof. Dr. Alexander Wolff
Dipl.-Inform. Philipp Kindermann

Homework Assignment #13
Computational Geometry (Winter Term 2014/15)

Exercise 1

The dual of a line segment is a left-right double wedge, as was shown in the lecture.

a) What is the dual of the set of points inside a given triangle with vertices p, q and r? **[3 points]**

b) What type of object in the primal plane would dualize to a top-bottom double wedge? **[3 points]**

Exercise 2

Let P be a simple regular polygon with n vertices. Let Δ_{blue} and Δ_{red} be triangulations of P, each given as a DCEL. We call the intersection points of edges from Δ_{blue} and Δ_{red} *Steiner vertices*. Let k be the number of Steiner vertices.

Give an $O(n + k)$ algorithm to compute a DCEL describing the *overlay* of Δ_{blue} and Δ_{red}, which consists of P, Δ_{blue}, Δ_{red} and the Steiner vertices. **[4 points]**

Exercise 3

Let R be a set of n red points in the plane, and let B be a set of n blue points in the plane. We call a line l a *separator* for R and B if l has all points of R to one side and all points of B to the other side.

a) Give a deterministic algorithm that can decide in $O(n \log n)$ time whether R and B have a separator. **[5 points]**

b) Give a randomized algorithm that can decide in $O(n)$ expected time whether R and B have a separator. **[5 points]**

Exercise 4

(Note: Based on a true incident.) A group of reserachers is working in a challenging problem in the field of information visualization. After finding some interesting results, the group decides to publish an academic paper about it. Immediately they start the writing to complete the paper before the deadline of a certain conference. Simultaneously, they implement an algorithm in Java to demonstrate the practical use of their theoretical work. Lazy as computer scientists are, they don't implement the algorithm from scratch, but rely on an existing library. They use the popular and „robust" *JTS Topology Suite* to compute a Delaunay triangulation. But the day before the deadline they are given a nasty shock! The algorithm gives wrong results. A suspicion quickly determines: *JTS Topology Suite* computes wrong Delaunay triangulations. What to do? They need your help to confirm the suspicion!

Download version 1.13 of the *JTS Topology Suite* from `http://sourceforge.net/projects/jts-topo-suite/`. Create a new Java project and copy the `com`-package (from `jts-1.13/src`) into the `src`-package of your project. Additionally, copy the java class `main.java` from WueCampus. This contains example code that computes a triangulation with the help of the *JTS Topology Suite* and displays them both textually and graphically.

a) Prove that the *JTS Topology Suite* Delaunay triangulation is wrong! To to that, give an instance (with at least three points) for which `dtb.getSubdivision()` computes an incorrect triangulation. (Hint: Only few points suffice!)
[5 extrapoints]

b) Why is the triangulation incorrect? Consider the source code of the implementation of `dtb.getSubdivision()`. Where can errors occur? (Hint: Take a look at the *frame*. Set `showFrame=true` to show the *frame*, colored in blue.)
[4 extrapoints]

c) Describe with words which steps would be necessary before calling `insertSites(vertices)` in `IncrementalDelaunayTriangulator.java`.
[3 extrapoints]

Please submit the first subexercise online. Just upload your edited file `main.java` in WueCampus.

This assignment is due at the beginning of the next lecture, that is, on January 21 at 10:15. Solutions will be discussed in the tutorial on Friday, January 23, 14:00–15:30 in room SE I.