Applied Computational Geometry — Spring 2009 — Dan Halperin

Assignment no. 2

due: March 30th, 2009

This is a warm-up exercise to get you acquainted with the CGAL library. You'll find additional useful information concerning usage of the library, example input files, submission of exercises and more in the course's website. Recall also that there is a help-desk where you can get assistance.

Exercise 2.1 Read an input file of line segments in the plane, compute the arrangement of the segments and display it on the screen, where the edges are drawn in blue. Find the face f of the arrangement which has the largest number of half-edges along its boundary and recolor the boundary edges of f red. If more than one face has the largest number—recolor the boundary of all of those faces. For simple drawing routines see the handouts of the lesson on March 16th, 2009.
The input file starts with an integer stating the number n of segments in the file, followed by n lines with four numbers each: x_1, y_1, x_2, y_2 for the coordinates of the endpoints of the segment. All the coordinate input numbers in one file are of the same type. A number can either be an integer, or a floating-point number and then it contains the decimal point, or it is a rational number written as a pair of integers separated by a slash with no spaces. Spaces separate the different numbers in the line. (Example input files can be found in the course's website.)

Exercise 2.2 Given an arrangement of segments (for example, as computed in Ex. 2.1), find the convex hull of the intersection vertices of the arrangement, namely of the points of intersection of the segments inducing the arrangement. Add the edges of the convex hull to the arrangement of the segments.

Exercise 2.3 Compute the number of half-edges on the boundary of the outer face in the arrangement computed in Ex. 2.2. Repeat this computation on several input files (which are given in the website), while using different number types

- machine `double`,
- `gmpq`,
- `CGAL::Quotient` with `MP_Float`,
- `CGAL::Quotient` with `gmpz`

geometry-traits classes

- `Arr_segment_traits_2<Kernel>`,
- `Arr_non_caching_segment_basic_traits_2<Kernel>`,
- `Arr_linear_traits_2<Kernel>`,
- (`Arr_circle_segment_traits_2<Kernel>`, ignoring "circles")

and kernels

- `Simple_cartesian<NT>`,
- `Cartesian<NT>`

Compare the results that you get and the running time.