CS 715 Computational Geometry

Line Segment Intersection

PROJECT REPORT

Aditya Gulati

This document gives a brief outline of the work done in the project. Details about the implementation have been skipped since they were discussed during the project demo. The code is available with this submission in any case.

1 The Problem

Given a set of n line segments, we want to find the k points of intersection.

This can trivially be done in $O(n^2)$ by checking all pairs. However, we are interested in an output sensitive algorithm that runs in less time.

2 The Algorithm - A Rough Sketch

We will use a sweep line algorithm. The sweep line moves from event point to event point, doing $O(1)$ operations at each point (checking intersection with neighbours for a particular line at each event point). The event points are start points, end points and intersection points. At any time, we have at most $n + k$ points in the sweep line status.

If we use a Balanced Binary Search Tree to maintain the status of the sweep line, we can perform all operations on the tree in at most $O(\log(n + k))$ time each. We will have to perform at most $O(n + k)$ such operations.

Thus, this algorithm will run in at most $O((n + k)(\log(n + k)))$.

3 The Implementation

Key features of the implementation of this algorithm are listed below:

- Runs in $O((n + k)(\log(n + k)))$ as expected.
- Takes input from a graphical interface. All input coordinates are therefore integers.
- The code is written in an Object Oriented manner. This makes it very easy to improve or extend functionalities. Faster data structures can be put in if needed (but this already runs optimally).

- To deal with degenerate cases, the points have been perturbed by a margin of 0.01. This is safe since all input points are at integer coordinates.
- The output is also graphical, showing the position of the sweep line at every stage.
- Input collected from the front end is put into a *txt* file and sent to the back end for processing. This file can be saved and passed to the back end too, independent of the front end thus giving the ability to save inputs making it easier to test.

Tools used:

- The code was written in python3.
- Tkinter was used to create the front end.
- SortedContainers were used as a substitute for BBSTs to store the sweep line status. All operations on these containers happen in $O(\log n)$. More details about SortedContainers can be found here.
- All classes for lines, points, collections etc. were written by me.

4 Running The Code

To run the program, simply run:

```
$ python3 frontEnd.py
```

You will see a blank canvas with 3 buttons. To draw the line segments:

- Click anywhere on the canvas. This will be the first end point of the line segment.
- Click again on the canvas. This will be the second endpoint of the line segment. The two endpoints will be connected automatically as soon as the second end point is placed.
- Repeat for multiple line segments.

Once all the line segments are drawn, hit the *Start Evaluation* button to start seeing the sweep line find the points. Intersection points along with end points of the line segments are marked in red.

To pass another input, hit the *Clear Screen* button to get a blank canvas.

In case you wanted to save this input, simply hit the *Save Input* button. It will be saved to the *data_files/examples* folder under the name "tc_num.txt", where *num* is a parameter supplied. It is initialised to 0 by default and incremented with every input that is saved.

To change the initial value, run the program as follows:

```
$ python3 frontEnd.py -n 2
```

To run a file from the examples folder, simply run:

```
$ python3 frontEnd.py --inp tc_2.txt
```

If you want to change the speed of the sweep line, it can be passed in miliseconds as follows:

```
$ python3 frontEnd.py --disp_delay 400
```

The code available comes with 9 testcases which can all be run by running the *run_tests.sh* file. Close a window to move to the next test case.

5 Dependencies

All the python libraries used are available with the default installation of python except sorted containers. To install this simply run:

```
$ pip install sortedcontainers
```