Algorithm Design, Anal. & Imp., Programming Assignment

Out: March 16, 2020, Due: April 20, 2020

Note:

- This assignment will carry 15 points towards your final score.
- You can use any of the following languages (C++, Java). Python language will be allowed only for those students whose undergraduate major was different than Computer Science.
- You can complete the assignment in a group of two students. You should form group by yourself. If you want to work individually, you are welcome to do so.
- You need to submit source code, which should run on a standard Linux machine. Your output should match exactly as outlined in this document.

Overlapping Rectangles

Background

VLSI databases commonly represent an integrated circuit (IC) as a list of rectangles. Typically, logic components are oriented rectilinearly (sides parallel to the x- and y-axis) on an IC; so, we can assume that all these rectangles are oriented accordingly. In such orientation, a rectangle can be represented by its minimum and maximum x- and y-axes. Now, an important problem in circuit design validation is to ensure that these rectangles do not overlap with each other. We can decide whether a pair of rectangles overlap in a constant time just by comparing the ranges of their x and y co-ordinates. If n rectangles are given, existence of an overlap among these can be decided in $\Theta(n^2)$ time. In this assignment, we like to find a better algorithm. We will use Interval tree to obtain a $\Theta(n \lg n)$ algorithm.

Implementation

You should implement a generic red-black tree class with a template parameter, which represents the value type. The class should have an insert and a delete method. Then you should use this class to write another class called Interval tree, which stores intervals, and supports interval insertion, deletion. Interval search method should also be implemented, which can report any overlap between an interval in the tree and a query interval. In class, we will show how we can map the rectangle

overlap problem to an interval search problem. Follow the method to report any overlap (if exist). Note that, you just need to report (any) one overlap.

Input

The input of the program is a Unix-format text file. Each line of the text file specifies the location of a rectangle. There are 5 numbers in each line which are separated by spaces. The first number is an integer which is the **id** of the rectangle, and the next 4 numbers are x_{low}, y_{low} and x_{high} and y_{high} respectively, where (x_{low}, y_{low}) and (x_{high}, y_{high}) are the bottom-left and top-right corner of the rectangle, so $x_{low} < x_{high}$ and $y_{low} < y_{high}$. The **id** of the rectangle matches the line number of the file. Rectangle on line 1 has id 1, rectangle on line 2 has id 2, and so on. You can assume that the input file in in correct format. Here is an example input file that contains 5 rectangles.

```
1 2.0 5.4 7.12 9.30
2 1.8 10.01 6.1 12.4
3 12.0 3.2 14.1 5.9
4 6.8 7.2 9.9 12.2
5 1.1 9.0 4.2 10.3
```

Output

If no overlap write on console, `no overlap`, otherwise write p q `overlap`, where p and q are ids of overlapping rectangle, and $p < q$. Note that if the sides of two rectangles shares a line, consider that as an overlap, too.

Collaboration and Usages of External Libraries

You are not allowed to use external libraries that implement any of the above data structures fully or partially. You are not allowed to collaborate with any other (except your project partner) inside or outside of the class. However, discussion of the project algorithm, idea, code and debug suggestion, posting of test data and output, etc. **in piazza forum are allowed and encouraged**. We will use software to detect similarity in submitted code, so please do not take change. The policy of academic dishonesty will be applied strictly, if evidence of dishonesty is found.

Deliverables

Please submit the source files in a tarzipped folder through canvas. The folder should be named as *IUPUI-id1_IUPUI-id2* and the submission file should be named as *IUPUI-id1_IUPUI-id2.tgz*. The

folder should have a README with instructions to compile and run the program. Program will be graded by automated scripts, so it is important that you follow the submission guideline carefully. TA will post detailed guideline about the organization of the source code after he has the grading script and test dataset ready, please follow the instruction. You may lose 15% of your score if you do not follow the posted guideline.