CS 577: Introduction to Algorithms 9/28/15

Homework 2 (Rubric)

Instructor: Dieter van Melkebeek TA: Bryce Sandlund

Problem 3 [10 points]

- [3 points] divide and conquer idea (break in the middle, recurse, and merge)
- [4 points] merge correctness
  - [1 point] consider segments crossing middle
  - [1 point] walking pointers moving out
  - [1 point] correct moving out conditions
  - [1 point] not missing any rectangles
- [3 points] analysis
  - [2 points] proving correctness
    - [1 point] merge procedure
    - [1 point] entire algorithm
  - [1 point] runtime argument

Many procedures had $O(n \log n)$ average case complexity, but $O(n^2)$ worst case complexity. For this, I deducted 4 points. Solutions that did not state the complexity was $O(n^2)$ in the worst case additionally lost a point for their runtime argument.

If an algorithm was incorrect, some assignments lost both points for correctness proofs, depending on the severity of the error.

Many people stated that their algorithm was correct for rectangles in the left half, right half, and crossing the middle, without explaining *why* their algorithm was correct in finding all possibly maximal rectangles that crossed the middle. I was looking for some mention of finding the maximal rectangle for a given height and that all possible heights crossing the middle are considered.

A number of solutions involved using a segment tree for range minimum query in $O(\log n)$ time. If argued correctly, I gave full credit, but it is nontrivial to argue the time complexity of a segment tree query, so many of you lost a point here. Furthermore, it's great that you learned a new, useful data structure, but I am aware most of you probably looked it up online. You won't be able to do that in an exam, so you should try to think of the purely divide and conquer $O(n \log n)$ solution.