CS302 - Problem Set 3

Please read the sections of the syllabus on problem sets and honor code before starting this homework.

1. Please write a formal proof for the correctness of the closest points algorithm using strong induction. This problem is entirely about proving correctness - you do not need to analyze the time complexity. You should combine all of the pieces we discussed in class into a proof that is easy to read and understand. You may use figures (pictures) in your proof, but you should clearly explain what is happening in the figure using full English sentences. The goal of this problem is to clearly and concisely explain complex mathematical/algorithmic ideas in English. I would recommend typing your proof so that it is easy to make edits. You should not turn in the first version you write - make sure you reread and make changes for clarity and correctness. For reference, my proof is about a page typed. In your proof, please refer to the following algorithm:

   `ClosestPair`(P) (where P is an array containing x-coordinates and y-coordinates of n points, where no two points have the same x- or y-coordinate.)

   Step 1: If $|P| \leq 3$ use brute force search and return closest distance.

   Step 2: Sort by x-coordinate into sets L and R

   Step 3: $\delta = \min\{$`ClosestPair`$(L),$ `ClosestPair`$(R)\}$

   Step 4: Create Y_δ, an array of points within δ of midline between L and R, sorted by y-coordinate.

   Step 5: Loop through elements of Y_δ, and calculate distance from each point to next 7 points, keeping track of δ', the smallest distance found.

   Step 6: Return $\min\{\delta, \delta'\}$

2. In class, we'll show that without a bit of cleverness, the recurrence relation for the runtime of the closest points algorithm is

$$T(1) = O(1), \qquad T(n) = 2T(n/2) + O(n \log n), \tag{1}$$

   but after some cleverness, we get a recurrence relation of

$$T(1) = O(1), \qquad T(n) = 2T(n/2) + O(n). \tag{2}$$

   (a) **[6 points]** What is the best bound possible on the first recurrence relation if we use the master method? Explain.

   (b) **[6 points]** Use the same strategy as we used to *create* the master method to analyze the first recurrence relation. Use big-O notation to bound your result. (See final page for some helpful reminders if you are getting stuck.)

   (c) **[3 points]** Now use the master method to analyze the second, improved recurrence relation. How much of an improvement did our cleverness get us?

3. For each of the following statements, if it is true, prove it, and if it is false, provide a counterexample and explanation.

   (a) **[11 points/6 points]** If a line graph has at least two vertices, the minimum-weight vertex is never part of the maximum-weight independent set.

   (b) **[11 points/6 points]** Considering the dynamic programming algorithm for MWIS on a line graph, if a vertex is excluded from the optimal solution of two consecutive subproblems, then it is excluded from the optimal solution to the final problem.

4. Approximately how long did you spend on this assignment (round to the nearest hour)?

For problem 2, recall that $\log(ab/c) = \log(a) + \log(b) - \log(c)$. Also $\log(2^k) = O(k)$. Also, $(\log a)^2 \neq 2\log(a)$. (The true equality is that $\log(a^2) = 2\log(a)$. Finally, you will probably want to look up the formula for evaluating an arithmetic series.