Problem Set 2—Due Monday, January 26 3:15PM

(20) Problem 1.

Do parts c, d, and g of Problem 4–1 (p. 85), and part a of Problem 4–4 (pp. 86).

(20) Problem 2. We look at analyzing a variant of the coin changing algorithms discussed in class (and posted on the web page). We consider the setting with pennies, dimes, quarters and half dollars.

(a) The following recurrence can be used to compute $C(n)$ the minimum number of coins to make change.

$$\begin{aligned} C(n) &= 1 \;\; n = 1,\; n = 10,\; n = 25,\; n = 50 \\ &= infinity \;\; n < 1 \\ &= 1 + MIN(C(n-1),\; C(n-10),\; C(n-25),\; C(n-50)) \end{aligned}$$

When using the *non-memoized* recursive algorithm to make change (implied by the above recurrence relation), let $W(n)$ be the number of calls to this recursive algorithm used to **compute** $C(n)$. Write a recurrence relation for $W(n)$. Note: you are **not** being asked to solve $W(n)$.

(b)

Let $M(n)$ denote the number of **calls** used to compute $C(n)$ using the *memoized* version of the algorithm. Argue that $M(n) = \Theta(n)$. Hint: do **not** use a recurrence relation.

(20) Problem 3. Consider a variation on the closest point algorithm we discussed (and in section 33.4). If there are 6 or fewer points we solve by brute force. Otherwise we split the points into **three** sets of (almost) equal size: P_L, the leftmost $n/3$ points, P_M, the middle $n/3$, and P_R the rightmost $n/3$. As before, we then recursively find the closest pair in each of the smaller sets (getting distances d_L, d_M, d_R), and then have to find the closest pair that crosses between two sets. As before you may assume we start by creating two arrays, X and Y with the points sorted by x and y coordinates respectively.

- **(a)** For this new setting, describe how to find any pairs that cross between two sets and are closer than the best distance found so far. You need not describe details that are essentially the same as for the original algorithm, but do carefully describe any changes. Try and implement this approach efficiently.(Warning: be careful to make sure you consider all possible cross-pairs that could beat the best distance found so far).
- **(b)** Formulate a recurrence relation which describes the worst case work for your algorithm of part a). (as before, you may ignore the work for the initial sort). Explain the terms in your recurrence relation.
- **(c)** Solve your recurrence relation of part b) to big Θ accuracy.

(15) Problem 4. Problem 4-3 part a), page 85.

(15) Problem 5. Consider the following proposed improvement to the closest point algorithm we discussed. Suppose we are doing the Combine step using two strips of width d (the closest distance found in P_L or P_R) and we find a pair of points that are $d' < d$ apart. The claim is that we can now adjust both strips to be of size d'. Thus, whenever we find a new, better distance in the Combine step, we adjust the strip width to this new distance.

- **(a)** Would the algorithm work correctly if we make this change? Justify your answer.
- **(b)** Describe how to change to a new (smaller) strip width during the Combine step.
- **(c)** Would this be a sensible change to make to the algorithm?

(15) Problem 6. Problem 5.2-5 on page 99.