CS715 Final Examination, May 15, 2008. No books or notes.

1. Find the solutions to the following recurrences. Prove each answer. You may use the following three statements without proof:

   - Let f' be the derivative of a function f.
     If $F(n) = F(n-1) + f'(n)$, then $F(n) = \Theta(f(n)$, for most f that you would be interested in.
   - If $F(n) = F(\log n) + 1$ then $F(n) = \Theta(\log^* n)$.
   - The Master Theorem.

   (a)

2. Give a definition of the A* algorithm, a proof of its correctness, and a discussion of when it is good to use it.

3. Give a very short formula that gives the volume of the convex hull of four given points in $\mathcal{R}^3$.

4. Describe in detail an algorithm that finds the closest pair of points in a set of n points in $\mathcal{R}^3$, 3-dimensional Euclidean space. What is the asymptotic time complexity of your algorithm?

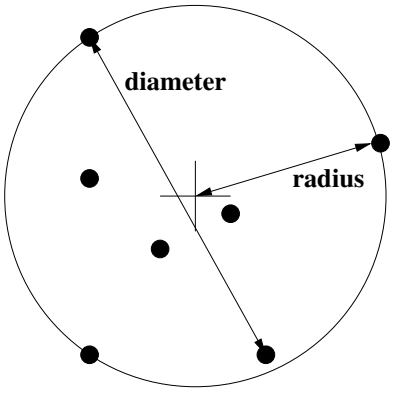
5. For any finite set of points X in a metric space M, we define the *diameter* of X to be the greatest distance between any two points of X, that is:

   $$diam(X) = \max_{\mathbf{x},\mathbf{y} \in X} ||\mathbf{x}, \mathbf{y}||$$

   where $||\mathbf{x}, \mathbf{y}||$ the distance between points $\mathbf{x}$ and $\mathbf{y}$.

   The *radius* of X is the radius of the smallest ball that encloses X, that is:

   $$radius(X) = \min_{\mathbf{r} \in M} \max_{\mathbf{x} \in X} ||\mathbf{x}, \mathbf{r}||$$

   You might think that $radius(X)$ is always half $diam(X)$, but this is false. Consider the case where M is the Euclidean plane. If $X = \{(1,0), (-1,0), (0,2)\}$, then $diam(X) = \sqrt{5}$ and $radius(X) = \frac{5}{4}$, which is more than half the diameter.

   - Give an $O(n \log n)$-time algorithm that computes the diameter of any set of n points in the Euclidean plane.
   - Given an algorithm that computes the radius of any set of n points in the Euclidean plane. State and prove the asymptotic time complexity of your algorithm.

6. Describe how RSA coding works. Include a description of how the encoding and decoding can be implemented efficiently. Be sure to state (not prove) any theorems that are used.

7. Draw a picture of the minimal DFA that detects whether a given input string contains a substring equal to the pattern string: [ ]

8. In order to make your task more straightforward, I have decided simplify the statement of the seaside traveler's problem. I first give a simple-minded "warmer" problem that is related.

   (a) We are given a triangular array $\{A[i,j] : 1 \leq j \leq i \leq n\}$, which has the following *reverse Monge* property: If $1 \leq j_1 < j_2 \leq i_1 < i_2$, then

   $$A[i_1, j_2] + A[i_2, j_1] \leq A[i_1, j_1] + A[i_2, j_2]$$

   Construct an $O(n \log n)$-time algorithm which finds all row minima of A.

   It is not known whether there is an $O(n)$-time algorithm for this problem. There is an existing algorithm which solves the problem in $O(n\alpha(n))$ time, where α is the inverse of Ackermann's function. This algorithm uses recursion that is phenomenally tricky.

   We did an $O(n \log n)$-time algorithm for the corresponding problem in the usual Monge case in class. Finding an $O(n \log n)$ for algorithm this problem is of about the same level of difficulty.

   (b) Let G be the weighted directed acyclic graph (V, E) where

   $$\begin{aligned} V &= \{0, 1, \ldots n\} \\ E &= \{(i,j) : 0 \leq i < j \leq n\} \\ \text{and } w(i_1, j_2) + w(i_2, j_1) &\leq w(i_1, j_1) + w(i_2, j_2) \end{aligned}$$

   for all $0 \leq i_1 < i_2 < j_1 < j_2$. Weights are allowed to be negative.

   Design an $O(n \log n)$ algorithm which computes a minimum weight path from 0 to n.

   Once again, this problem can be solved in $O(n\alpha(n))$ time. The corresponding problem where the inequality is reversed can be solved in $O(n)$ time.

9. More problems.