CSE 521: Design and Analysis of Algorithms — Homework #1

The stuff you should already know. — March 31, 2009.

Due: April 9, 2009. — Reading: Kleinberg-Tardos, pages 1–335.

The problems are worth 10 points each. If I ask you to write down an algorithm, use pseudocode.

1. **Asymptotic analysis.** Sort the following functions from asymptotically smallest to largest, indicating ties if there are any:

   $n, \log n, \log\log^* n, \log^* \log n, \log^* n, n\log n, \log(n \log n), n^{n/\log n}, n^{\log n}, (\log n)^n, (\log n)^{\log n}, (1+\frac{1}{n})^n$

   $2^{\sqrt{\log n \log\log n}}, 2^n, n^{\log\log n}, n^{1/1000}, (1+\frac{1}{1000})^n, (1-\frac{1}{1000})^n, (\log n)^{1000}, \log_{1000} n, (\log 1000)^n, 1$

   [To simplify notation, write $f(n) \ll g(n)$ to mean $f(n) = o(g(n))$ and $f(n) \equiv g(n)$ to mean $f(n) = \Theta(g(n))$. For example, the functions $n^2, n, \binom{n}{2}, n^3$ are sorted as $n \ll n^2 \equiv \binom{n}{2} \ll n^3$.]

2. **Linearity of expectation.** Suppose that $x_1, x_2, \ldots, x_n \in [0,1]$ are chosen uniformly and independently at random. We are going to analyze a very simple sorting algorithm which sorts the numbers $\{x_1, \ldots, x_n\}$ in $O(n)$ *expected time.*

   There are going to be n buckets $B_1, B_2, \ldots, B_n$. For a real number x, we use $\lceil x \rceil$ to denote the smallest integer greater than x. The algorithm is as follows.

   (a) For $i = 1, 2, \ldots, n$ put x_i into bucket B_j where $j = \lceil x_i \cdot n \rceil$.

   (b) For $j = 1, 2, \ldots, n$ sort B_j.

   (c) Concatenate the sorted buckets.

   Part 1: Give a *brief* description of how you would implement the steps of the algorithms so that the total running time is

   $$O(n) + \sum_{j=1}^{n} O(|B_j|^2).$$

   Part 2: Show that the *expected* running time (over the random choice of inputs) of your algorithm is $O(n)$.

3. **Dynamic programming.** Consider two strings X and Y over the alphabet $\{A, C, G, T\}$. The *edit distance* between X and Y is the minimum cost of a sequence of edit operations which turns X into Y. The operations are as follows.

   (a) Insert a character (cost 2).

   (b) Delete a character (cost 2).

   (c) Replace a character (cost 1).

   Design and formally analyze an algorithm for computing the edit distance (i.e. the *minimum cost*) between X and Y which runs in time $O(|X| \cdot |Y|)$. Here, $|X|$ denotes the length (number of characters) in the string X.

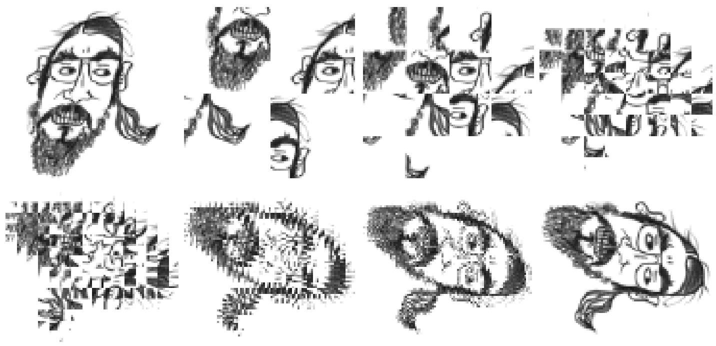

Figure 1: The first algorithm (blit then recurse) in action.

4. **Divide and conquer** (borrowed from Jeff Erickson). Some graphics hardware includes support for an operation called *blit,* or **bl**ock **t**ransfer, which quickly copies a rectangular chunk of a pixelmap (a two-dimensional array of pixel values) from one location to another. This is a two-dimensional version of the standard C library function memcpy().

   Suppose we want to rotate an $n \times n$ pixelmap 90° clockwise. One way to do this is to split the pixelmap into four $n/2 \times n/2$ blocks, move each block to its proper position using a sequence of five blits, then recursively rotate each block. Alternately, we can first recursively rotate the blocks and then blit them into place afterwards. See Figures 1 and 2.

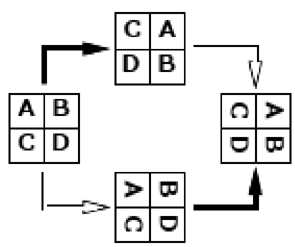

Figure 2: Two algorithms for rotating a pixelmap. Black arrows indicate blitting the blocks into place. White arrows indicate recursively rotating the blocks.

   In the following questions, assume n is a power of two.

   (a) Prove that both versions of the algorithm are correct.
   (b) *Exactly* how many blits does the algorithm perform?
   (c) What is the algorithm's running time if a $k \times k$ blit takes $O(k^2)$ time?
   (d) What if a $k \times k$ blit takes only $O(k)$ time?

5. **Graph algorithms.** Write an algorithm that, given an undirected graph $G = (V, E)$ in adjacency list representation, detects whether G contains a cycle. Your algorithm should run in $O(m + n)$ time where $m = |E|$ and $n = |V|$.