Problem Set 1

This problem set is due in class on *Thursday, Febuary 15th* at the beginning of class.

There are four problems. **Each problem is to be done on a separate sheet (or sheets) of paper. Separate problems should *not* be stapled together.** Mark the top of **each and every sheet** with your name, 6.046J/18.410J, the **problem number**, your **recitation section, your TA's name**, and the date. It is very important to hand in problems separately and to indicate your recitation section **and** TA name on every sheet. Failure to do so will result in delays in returning your work and possible confusions with entering your grade. All problem sets throughout the semester should be handed in this way.

Reading: Chapters 1–2 of CLR.

Problem 1-1. Ordering Functions

Rank the following functions by order of growth; that is, find an arrangement $g_1, g_2, \ldots, g_{30}$ of the functions satisfying $g_1 = \Omega(g_2)$, $g_2 = \Omega(g_3)$, $\ldots$, $g_{29} = \Omega(g_{30})$. Partition your list into equivalence classes such that $f(n)$ and $g(n)$ are in the same class if and only if $f(n) = \Theta(g(n))$. Note that $\lg x$ refers to the base two logarithm.

$$\begin{array}{ccccc}
n \lg n & \lg(n!) & 4^{\lg n} & n & \lg^* n \\
\left(\frac{3}{2}\right)^n & \left(\frac{2}{3}\right)^n & n^3 & n^2 + n^3 & \left(\sqrt{2}\right)^{\lg n} \\
1 & (\lg n)! & n^{\lg \lg n} & n^{\lg n} & 2^{\lfloor \lg n \rfloor} \\
2^{\lceil \lg n \rceil} & \lg n^2 & n! & (n-1)! & (2n)! \\
\lg \lg n & \ln n & 2^{2^n} & e^n & 2^{2n}
\end{array}$$

Problem 1-2. Asymptotic Notation

For each of the following statements, decide if it is **always true**, **never true** or **sometimes true** for non-negative functions $f(\cdot), g(\cdot)$. If it is **always true** or **never true**, explain why. If it is **sometimes true**, give one example for which it is true, and one for which it is false.

(a) $f(n) = O(f(n)^2)$

(b) $f(n) \neq O(g(n))$ and $g(n) \neq O(f(n))$

(c) $f(n) + g(n) = \Theta\left(\max\left(f(n), g(n)\right)\right)$

(d) $f(n) + o(f(n)) = \Theta(f(n))$

(e) $f(n) = \Omega(g(n))$ and $f(n) = o(g(n))$ (note the little-o).

Problem 1-3. Insertion sort on small arrays in merge sort

Although merge sort runs in $\Theta(n \lg n)$ worst-case time and insertion sort runs in $\Theta(n^2)$ worst-case time, the constant factors in insertion sort make it faster for small n. Thus, it makes sense to use insertion sort within merge sort when subproblems become sufficiently small. Consider a modification to merge sort in which n/k sublists of length k are sorted using insertion sort and then merged using the standard merging mechanism, where k is a value to be determined.

(a) Show that the n/k sublists, each of length k, can be sorted by insertion sort in $\Theta(nk)$ worst-case time.

(b) Show that the sublists can be merged in $\Theta(n \lg(n/k))$ worst-case time.

(c) Given that the modified algorithm runs in $\Theta(nk + n \lg(n/k))$ worst-case time, what is the largest asymptotic (Θ-notation) value of k as a function of n for which the modified algorithm has the same asymptotic running time as standard merge sort?

(d) How should k be chosen in practice ? (I.e., for a specific implementation, where the constants associated with the running time can be calculated).

Problem 1-4. Recurrences

Solve the following recurrences. Assume $T(1) = 1$ in all cases except (c) where you can assume $T(3) = 1$. Give your answer in Θ notation. Justify your answers.

(a) $T(n) = 4T(\frac{n}{2}) + n^3$

(b) $T(n) = T(\frac{n}{2}) + T(\frac{n}{3}) + n$

(c) $T(n) = 3T(n^{\frac{1}{3}}) + \log_3 n$

(d) $T(n) = T(n-1) + n^4$

(e) $T(n) = T(n/2 + 5) + n^2$.

(f) $T(n) = 2T(n/2) + n \lg n$.

(g) $T(n) = 4T(n/2) + n^2 + n$.