Problem Set 2
IE 496 – Computational Methods in Optimization
Dr. Ralphs

1. Give asymptotic upper and lower bounds for each of the following recurrences. As usual, you may assume that $T(n)$ is constant for sufficiently small values of n.

   (a) $T(n) = 2T(n/2) + n^3$

   (b) $T(n) = 2T(n/4) + \sqrt{n}$

   (c) $T(n) = T(\sqrt{n})$

   (d) $T(n) = 4T(n/2) + n^2\sqrt{n}$

   (e) $T(n) = T(n-1) + \log n$

2. We generally assume that arrays are *passed by reference*, since this is the norm in most modern languages. This problem examines the implications on the running times of recursive algorithms when arrays are passed by copying. We consider recursive implementations of both binary search and merge sort. Analyze these algorithms for the following cases:

   (a) The array is passed by reference in each recursive call, i.e., the cost of each call is $\Theta(1)$.

   (b) The array is passed by copying the entire array each time, i.e., the cost of each call is $\Theta(n)$.

   (c) The array is passed by copying only the relevant subarray each time, i.e., the cost of each call is $\Theta(q - p + 1)$ when the subarray $A[p..q]$ is passed.

3. Implement merge sort recursively in both Python and C++ and verify the theoretical analysis from the previous problem empirically. Compare the two implementations to each other as well as to a a nonrecursive version of merge sort.

4. Discuss how you might implement merge sort in parallel and analyze the running time of the parallel algorithm theoretically, assuming a PRAM model of computation.