CS 473: Algorithms, Fall 2009
HBS 1

1. **Recurrences:**

   (a) Solve asymptotically: $A(n) = 2 \cdot A(n/2) + n \log n$.

   (b) Solve asymptotically: $B(n) = B(5n/8) + B(n/4) + n$.

   (c) Suppose you have two strategies to solve a problem: you can either divide it into 3 subproblems each of size $n/2$ or divide it into 2 subproblems each of size $n/3$. If the work to combine subproblems is n in both cases, what is a better solution? Which is a better solution if the work to combine subproblems is n^2?

2. Suppose you are given k sorted arrays $A_1, A_2, \ldots, A_k$ where each array contains n elements. The goal is to merge all the arrays into a single sorted array A of kn elements. Given two sorted arrays of size s and t respectively, you know that they can be merged into a single sorted array in $O(s + t)$ time.

   (a) Suppose you use the following algorithm for merging the k arrays. Merge A_1 and A_2. Merge the resulting array with A_3 and the result with A_4 and so on. What is the running time of this algorithm as a function of k and n?

   (b) Give a more efficient algorithm using divide and conquer.

   (c) Consider the following modification to the merge sort algorithm. Instead of splitting the input array into 2 subarrays, recursively sorting each and merging the 2 sorted subarrays, we will split the input array into k subarrays, recursively sort each (using the modified algorithm), and merge the k sorted subarrays. How does the running time of the modified algorithm compare to that of the original algorithm?