

CMPS 102 Homework 4

4 problems, 25 points, due at the start of class, May 3, 2005

Reading: Chapter 8, and the reading from HW 3.

1. (10 pts) A 101 assignment requires the implementation of a hash table. One of the students in the class thinks that by making the hash table big enough, the chance of collision will be negligible, and thus the bug in his linked list will go unnoticed. Since the program is given $k > 2$, the number of elements to be inserted ahead of time, he decides to have his hash table contain k^2 locations.
 - (2 pts) How many different ways can the k different elements, x_1, \dots, x_k be hashed into the k^2 table locations?
 - (3 pts) How many of these ways result in no collisions? (you may want to use factorials for this answer).
 - (1 pts) If each way has the same probability, then what is the probability of one or more collisions? (You may want to use the factorials in this answer).
 - (1 pt) Use a calculator to compute this probability for $k = 4$.
 - (3 pts) Let $X_{i,j}$ be the indicator variable that is 1 when keys x_i and x_j collide and 0 when keys x_i and x_j get hashed to different bins. Assume that each different way of hashing the k elements into the table has the same probability and use these indicator variables to compute the expected number of collisions (the expected number of pairs i, j such that x_i and x_j get hashed into the same bin – if keys x_1, x_2, x_3 all get hashed into the same bin then that bin has three collisions: x_1, x_2, x_1, x_3 , and x_2, x_3 .)
2. (5 pts) Assume $n = 2^k$ and find a way to multiply two (positive) n -bit numbers using three multiplications of $n/2$ -bit numbers. Describe (in pseudo-code) and analyze a recursive divide-and-conquer algorithm for multiplication and analyze its running time. You can assume that the computer has a “shift” instruction that takes 1 unit of time (see page 22), and adding two n -bit numbers takes $\Theta(n)$ time. To simplify the analysis, assume that no “carry-outs” happen, so the addition of two ℓ -bit numbers always gives a result fitting in ℓ -bits.
3. (4 pts) Consider a modification to MERGESORT that recursively calls itself at most k times and then uses Selection Sort (if needed) to sort the 2^k resulting sub-arrays (each of these sub-arrays has approximately $n/2^k$ elements). Thus the recursion tree for this algorithm has k levels of MergeSort followed by a level of Selection Sort.

Determine the asymptotic running time of the algorithm as a function of n when k is a constant. You may assume that n is a power of 2, and that running Selection Sort on n elements takes $\Theta(n^2)$ time.

4. (6 pts) Prove the correctness of Radix Sort (pg 172 of text) by (finite) induction on i , the column being sorted. It should be clear from your proof why the sorting method for each digit must be stable, and why the digits must be sorted least to most significant.