Computer Science 373 – Analysis of Algorithms
Prof. Steven Skiena
Fall 2019

Daily Homework Problems

This is the schedule of daily homework problems for the semester. Each must be turned in at the beginning of the given class. I will start class by presenting a solution to the daily problem. Seriously working on the daily problems, and confirming that you understand the solution is perhaps the single most important way to learn the material in this course.

8/29 Problem 1-5.

9/3 Problem 2-22.

9/5 Problem 2-20.

9/10 For each of the four types of linked lists in the following table, what is the asymptotic worst-case running time for each dynamic-set operation listed?

	singly unsorted	singly sorted	doubly unsorted	doubly sorted
Search(L, k)				
Insert(L, x)				
Delete(L, x)				
Successor(L, x)				
Predecessor(L, x)				
Minimum(L)				
Maximum(L)				

9/12 You are given the task of reading in n numbers and then printing them out in sorted order. Suppose you have access to a balanced dictionary data structure, which supports each of the operations search, insert, delete, minimum, maximum, successor, and predecessor in $O(\log n)$ time.

 – Explain how you can use this dictionary to sort in $O(n \log n)$ time using only the following abstract operations: minimum, successor, insert, search.

 – Explain how you can use this dictionary to sort in $O(n \log n)$ time using only the following abstract operations: minimum, insert, delete, search.

 – Explain how you can use this dictionary to sort in $O(n \log n)$ time using only the following abstract operations: insert and in-order traversal.

9/17 Problem 4-3.

9/19 Give an efficient algorithm to determine whether two sets (of size m and n) are disjoint. Analyze the complexity of your algorithm in terms of m and n. Be sure to consider the case where m is substantially smaller than n.

9/24 The *nuts and bolts* problem is defined as follows. You are given a collection of n bolts of different widths, and n corresponding nuts. You can test whether a given nut and bolt together, from which you learn whether the nut is too large, too small, or an exact match for the bolt. The differences in size between pairs of nuts or bolts can be too small to see by eye, so you cannot rely on comparing the sizes of two nuts or two bolts directly. You are to match each bolt to each nut.

1. Give an $O(n^2)$ algorithm to solve the nuts and bolts problem.
2. Suppose that instead of matching all of the nuts and bolts, you wish to find the smallest bolt and its corresponding nut. Show that this can be done in only $2n - 2$ comparisons.
3. Match the nuts and bolts in expected $O(n \log n)$ time.

10/3 Problem 5-8.

10/8 Problem 5-4.

10/10 Problem 5-23.

10/17 Problem 6-6.

10/22 Problem 6-15.

10/24 Problem 6-22.

10/29 Problem 7-1.

10/31 Problem 7-2.

11/5 Problem 8-2.

11/7 Problem 8-12.

11/12 Problem 8-15

11/14 We wish to compute the laziest way to dial given n-digit number on a standard push-button telephone using two fingers. We assume that the two fingers start out on the * and # keys, and that the effort required to move a finger from one button to another is proportional to the Euclidean distance between them. Design and analyze an algorithm that computes in time $O(n)$ the method of dialing that involves moving your fingers the smallest amount of total distance.

11/26 Problem 9-3.

12/3 Problem 9-10.

12/5 Problem 9-13.