Basic Algorithms (Section 7) Practice Midterm
Instructor: Jiaxin Guan

October 14, 2024

1. This is a practice exam to help you prepare. You do not need to turn it in.
2. You have 75 minutes to complete the real midterm, this exam is intended to be comparable in length and difficulty.
3. You may use a double-sided, letter-size "cheatsheet". The use of phones, computers, or other reference material during the exam is not permitted.
4. You may use any algorithm or theorem we saw in class (or homework) without proof, as long as you state it correctly. For all questions asking you to give algorithms, you do not have to give detailed pseudo-code, or even any pseudo-code. It is enough to give a clear description of your algorithm. You should additionally give a brief justification of the correctness and the claimed run time of your proposed algorithm.
5. The structure of the real midterm will be somewhat similar to what is below, but I might adjust the number of multiple choice/algorithm design questions. In general, you should expect questions of the following types:
   (a) Questions that can be solved if you understood what was presented in class (definitions, how the algorithms work).
   (b) Questions that can be solved by reducing to something we did in class (e.g. use some algorithm from class to solve a new problem).
   (c) Questions that can be solved by adapting an idea from class (e.g. Question 2 in HW5 asks you to modify the rod cutting DP algorithm to handle unique lengths).
6. This exam contains 5 pages (including this cover page).

Question	Points	Score
Multiple Choices	40	
k Largest Elements	25	
Longest Increasing Subsequence	35	
Total:	100	

1 Multiple Choices (40 points)

Choose the best answer for each of the questions below. No justification is needed.

1. Which of the following is **not** $O(n \log n)$? .... (   )
   (A) $\frac{9n^2}{(\log n)^3}$
   (B) $\sqrt{58n} \cdot (\log n)^2$
   (C) $87n^{\frac{\log_5 n}{\log_3 n}}$
   (D) $n \cdot \log(n^3)$

2. Let $f(n), g(n), h(n)$ be positive functions. Which one of the following statements is true? .... (   )
   (A) If $f(n) \neq o(g(n))$, then $f(n) = \Omega(g(n))$
   (B) If $f(n) = o(g(n))$ and $g(n) = \omega(h(n))$, then $f(n) = \Theta(g(n))$
   (C) If $f(n) = O(g(n))$, then $(f(n))^2 = O((g(n))^2)$
   (D) None of the above

3. Suppose we have $T(n) = 2T(n/3) + n$ and $T(0) = T(1) = 1$, which of the following statements is false? .... (   )
   (A) $T(n) = \omega(n^{2/3})$
   (B) $T(n) = \Omega(n \log n)$
   (C) $T(n) = O(n \log n)$
   (D) $T(n) = o(n^2)$

4. Which of the following is **not** an example of a divide-and-conquer algorithm? . (   )
   (A) QuickSort algorithm for sorting an array
   (B) Karatsuba's algorithm for fast integer multiplication
   (C) Deterministic Selection algorithm for finding the median
   (D) Gale-Shapley algorithm for stable matching

5. Consider the QuickSort algorithm where we always pick the last element as the pivot and arrays $A = [1, 2, 3, \ldots, n]$ and $B = [n, n-1, n-2, \ldots, 1]$. Let C_A be the number of comparisons made when running QuickSort on A, and C_B be the number of comparisons made when running QuickSort on B. Then we have .... (   )
   (A) $C_A > C_B$
   (B) $C_A = C_B$
   (C) $C_A < C_B$
   (D) Cannot say anything for arbitrary n

6. Alice comes up with a comparison-based sorting algorithm whose best-case run time is $O(n)$, and Bob comes up with a comparison-based sorting algorithm whose worst-case run time is given by the recurrence $T(n) = 4T(n/5) + 5n \log \log n$. Which of these two algorithms can **possibly** be correct? .... (   )

(A) Neither Alice's nor Bob's
(B) Only Alice's
(C) Only Bob's
(D) Both Alice's and Bob's

7. The subset-sum problem is defined as follows. Given a set of n positive integers $A = \{a_1, a_2, a_3, \ldots, a_n\}$ and positive integer s, is there a subset of A whose elements sum to s?

   A DP algorithm for solving this problem uses a 2-dimensional Boolean array X, with n rows and $s+1$ columns. The entry $X[i,j]$ for $1 \leq i \leq n, 0 \leq j \leq s$ is TRUE if and only if there is a subset of $\{a_1, a_2, \ldots, a_i\}$ whose elements sum to j.

   With that in mind, which of the following is valid for $2 \leq i \leq n$ and $a_i \leq j \leq s$? ($\wedge$ is the logical AND, and $\vee$ is the logical OR) .... (   )

   (A) $X[i,j] = X[i-1,j] \vee X[i, j-a_i]$
   (B) $X[i,j] = X[i-1,j] \vee X[i-1, j-a_i]$
   (C) $X[i,j] = X[i-1,j] \wedge X[i, j-a_i]$
   (D) $X[i,j] = X[i-1,j] \wedge X[i-1, j-a_i]$

8. Which of the following statements is true? .... (   )

   (A) For the same problem, the DP algorithm with memoizaiton/bottom-up always performs better than a Divide-and-Conquer algorithm
   (B) To justify the correctness of a DP algorithm, it is sufficient to justify the correctness of the optimal substructure used in the algorithm
   (C) There can be more than one possible optimal substructure for a DP problem
   (D) None of the above

2 k Largest Elements (25 points)

Describe an $O(n)$ time algorithm that, given an unordered array of n arbitrary integers, and an integer $k \in \{1, \ldots, n\}$, outputs the k largest integers in the array (not necessarily in order). Justify the correctness and run-time of your proposed algorithm.

3 Longest Increasing Subsequence (35 points)

Given a sequence of integers of length n, we want to find the length of the longest increasing subsequence, where the elements monotonically increases. For example, if the input sequence is $1, 7, 4, 5, 8, 3, 9, 6, 2$, then $1, 4, 8$ is such a sequence, as well as $7, 8, 9$ and $4, 5, 6$.

(a) What is the length of the longest increasing subsequence for the example $1, 7, 4, 5, 8, 3, 9, 6, 2$? And what is the subsequence that yields this length? (5 points)

(b) Design an $O(n^2)$ algorithm that finds the length of the longest increasing subsequence. You only need to output the length, not the sequence itself. For simplicity, assume all input numbers are distinct, i.e. no repetitions. Justify the correctness and run-time of your proposed algorithm. (25 points)