Exam in Algorithms and Advanced Data Structures (1DV516)

2022-12-09

The exam consists of 8 problems, each worth 10 points. The questions are not arranged in order of difficulty. You may answer in Swedish or English. Illegible answers are not corrected!

- Grade A: 71 - 80 points
- Grade B: 61 - 70 points
- Grade C: 54 - 60 points
- Grade D: 47 - 53 points
- Grade E: 40 - 46 points

When an algorithm is asked for, it should be understood that it should be as efficient as possible, and it should be expressed in such a way that it is understandable. When asked about complexity, explain what N is or be as specific as possible (e.g., $O(|V|)$ is better than $O(N)$).

Problem 1 (4 + 2 + 2 + 2 = 10 p)

1. The asymptotic run time (complexity) of Quicksort depends on the choice of the Pivot value. Assume you pick the first (left-most) value. What is the average and worst case for Quicksort when the input is in random order? What if the input is already sorted in ascending order? Motivate.
2. Draw the recursive calls required to sort the array $[13, 24, 31, 43, 49, 57, 69]$ with Mergesort
3. What is the worst case asymptotic run time (complexity) of the Merge? Motivate.
4. What is the worst case asymptotic run time (complexity) of the Mergesort? Motivate.

Problem 2 (3 + 3 + 4 = 10 p)

1. Order these functions in order of asymptotic growth rate, from fastest growth to slowest: $\log_2 N^2$, $\log_2 N$, 2^{2^N}, 4^N, $0.000001N$, $N \log_2 N$, and $\log_2 2N$. Indicate if two functions have the same growth rate. Briefly motivate your ordering.

2. Prove or disprove: $N^3 + 3N^2 + 11N + 3 = O(N^4)$. Note, you must use the theory presented in class; it is not sufficient to reason about the exponents.

3. An algorithm takes 0.75 ms for input size 100. How long will it take for input size 800 if the running time is the following (assume low-order terms are negligible): linear, $O(N \log N)$, quadratic, cubic, and exponential (2^N)?

Problem 3 (2 + 2 + 3 + 3 = 10 p)

1. Suppose that we have numbers between 1 and 100 in a binary search tree and want to search for the number 45. Which (possibly multiple) of the following sequences could be the sequence of nodes examined?

    1. 1, 2, 3, 4, 5, 6, 7, 8.
    2. 5, 2, 1, 10, 39, 34, 77, 63.
    3. 8, 7, 6, 5, 4, 3, 2, 1.
    4. 9, 8, 63, 0, 4, 3, 2, 1.
    5. 50, 25, 26, 27, 40, 44, 42.
    6. 50, 25, 26, 27, 40, 44.

2. What is the smallest, largest , and expected (average) height of a Binary Search Tree with N elements? Why?

3. The BST-sort algorithm looks at each element of the array in turn, starting at position 0, and inserts it into a BST (pass 1). Having processed all elements, it repeatedly extracts the minimum from the BST, refilling the array from position 0 onwards (pass 2). Provide pseudo code for BST-sort and give the asymptotic growth (complexity) for the two phases. Motivate.

4. How can Phase 2 of BST-sort be improved without changing the data structure? What is the symptotic growth (complexity) of the improved Phase 2 and BST-sort?

Problem 4 (2 + 4 + 4 = 10 p)

1. Given any directed graph $G = (V, E)$ with non-negative edge weights, consider the problem of all-pairs shortest path (APSP). Give the asymptotic runtimes of Bellman-Ford and Dijkstra's algorithms when applied to the APSP problem as a function of $|V|$ and $|E|$, and provide a brief justification for your answer.

2. Does Dijkstra's algorithm work on graphs with negative edge-weights? Explain why or why not.

3. Explain why Bellman-Ford works with negative edge-weights but not negative cycles.

Problem 5 (5 + 1 + 4 = 10 p)

1. Draw the AVL tree that results from inserting the keys: 1, 3, 7, 5, 6, 9 in that order into an initially empty AVL tree. Show and explain each step.
2. Draw the Splay tree that results from inserting the keys: 5, 7, 9, 6, 1, 3 in that order into an initially empty Splay tree.
3. Draw the Splay tree that results from searching for the keys: 3, 6, 9 in that order in the Splay tree created in 5.2. Show and explain each step.

Problem 6 (3 + 3 + 4 = 10 p)

1. Explain memoization.
2. If a problem can be solved by dynamic programming or a greedy algorithm, what are the advantages of each approach? Which one would you prefer to use? Why?
3. Write a memoized recursive function, `F(n: int) -> int`, to compute the nth Fibonacci number ($n > 0$). Recall that $F_1 = 1, F_2 = 1$, and $F_n = F_{n-1} + F_{n-2}$. You can use psuedo code, Java, or Python.

Problem 7 (5 + 5 = 10 p)

An imaginary post office machine must issue stamps adding up to a given amount of n SEK. Its goal is to minimize the number of postage stamps issued, and the machine always has as many stamps as needed.

1. Let the set of available denominations for the stamps be $D = \{0.01, 0.05, 0.25, 0.50, 1, 2\}$. Can this problem be solved using bottom-up dynamic programming? If so, clearly describe your algorithm and determine its complexity. If not, show why it cannot be done.
2. Provide a set of denominations for stamps D and an amount of n SEK for which the greedy strategy fails to give an optimal solution, n being a multiple of the smallest denomination in D. Show what solution the greedy strategy would find and what the optimal solution is.

Problem 8 (5 + 5 = 10 p)

Assume that we have the set $S = \{14, 23, 32, 41, 50, 59, 68\}$ and we want to insert them into a hash table T of size at most 10, using chaining to resolve collisions.

1. Provide a size of T and a suitable hash function h, such that the distribution of elements in T by using h would be good for random input but bad for the elements in S.
2. Provide a size of T and a suitable hash function h, such that the distribution of elements in T by using h would be good for random input and for the elements in S.