CS2040S: Data Structures and Algorithms

Discussion Group Problems for Week 10

For: Oct. 24, 2019

Problem 1. Given the array A, you have to find the length of the longest *consecutive* sequence that you can create from the numbers of A. For example:

$A = [3, 8, 5, 3, 12, 13, 1, 3, 12, 2, 12, 4]$, then the longest sequence you can create is $S = [1, 2, 3, 4, 5]$, so your method should output 5. Note: the numbers in S were not originally sorted in A.

Describe the most time-efficient algorithm you can think of to find the length of the longest consecutive sequence you can create.

Problem 2. Spy Net

You are 00-42; a super spy that has infiltrated the lair of the evil Dlorah Hos. You have discovered that she plans to secretly replace the CS2040 final exam questions with *ultra difficult problems*! How evil! You aim to prevent this from happening by sending a message to NUS through a computer network $G = (V, E)$.

Sending a message across each link $e = (u, v) \in E$ takes $t(u, v)$ seconds. As such, the total time to send a message on a path $p = u \to v \to w \to ... \to y \to z$ takes $T(p) = t(u, v) + t(v, w) + \cdots + t(y, z)$. Your goal is to send the message as quickly as possible from a source node s to destination node d.

Problem 2.a. Describe a time-efficient algorithm to find the fastest path from s to d through the computer network, i.e., the path p with minimal $T(p)$.

Intercepted! Uh oh! Your initial message was intercepted! Looks like Dlorah's agents are on high-alert for potential leaks so, any message sent through the network risks interception while in transit.

Each link $e = (u, v) \in E$ is also associated with a security score $s(u, v)$. You can interpret the score $s(u, v)$ as the probability that your message will remain secure (i.e., it will not be intercepted). As such, $s(u, v)$ is between 0 and 1, that is, $0 \leq s(u, v) \leq 1$.

Assume that these edge probabilities are independent. Then, for any given path through the computer network $p = u \to v \to w \to ... \to y \to z$, the security of the entire path p is the product of the security scores of all the edges on the path: $S(p) = s(u, v) \times s(v, w) \times \cdots \times s(y, z)$. You want to send your message via the most secure path, i.e., the path p which has *maximum* $S(p)$.

Problem 2.b. Describe the most time-efficient algorithm you can think of to find the most secure path through the computer network.

Problem 3. Leaves For Me

In many applications, the elements of interest are stored as leaves. As such, it is often necessary

to get a list of the leaves in some order. This problem asks you to solve this problem for two different data structures.

Problem 3.a. Describe an efficient algorithm that prints out the leaves in a Binary Heap in *descending order*.

Problem 3.b. Describe an efficient algorithm that prints out the leaves in a BBST in *descending order*.

Problem 4. Distributed Computation (Optional)

You have been recently hired by a SNS, a sneaky new startup that aims to leech processing time from many small devices connected to its website (e.g., laptops, mobile phones) to mine BitDollars. The key problem is to distribute work packages to these devices and combine them in an intelligent way. One of their subproblems requires the sorting of a array, and they've figured out the how to distribute the work, but not how to combine the results.

Problem 4.a. Describe the most time efficient algorithm you can think of for combining k sorted arrays into a new sorted array (ascending order).

Problem 4.b. Describe the most time efficient algorithm you can think of for combining k BSTs.

Problem 5. Palindromic Permutations (Optional)

Palindromes are the words that read the same forwards or backwards, e.g., "racecar", "anna", "level", "rotator" or "madam". For this problem, we want to know if a given string S can be *permuted* to form a palindrome. For example, "edified" can be permuted to form "deified", which is a palindrome. As other examples, "carerac" can be permuted to form "racecar", and "nev-erevenorodd" can be permuted to become "neveroddoreven". For this problem, assume that the string is a single word without spaces or punctuation. The original and permuted strings need not be valid English words.

Problem 5.a. Describe a time-efficient algorithm for figuring out if the letters in a given string S can be *permuted* to form a palindrome.

Problem 5.b. Given a string S that may or may not be a permutation of a palindrome. Describe a time-efficient algorithm to compute the length of the longest palindrome that can be formed using characters from S.