ShanghaiTech University

CS240 Algorithm Design and Analysis

Fall 2021

Problem Set 1

Due: 23:59, Sept. 30, 2021

1. Submit your solutions to Gradescope (www.gradescope.com).
2. In "Account Settings" of Gradescope, set your FULL NAME to your Chinese name and enter your STUDENT ID correctly.
3. If you want to submit a handwritten version, scan it clearly. Camscanner is recommended.
4. When submitting your homework, match each of your solution to the corresponding problem number.

Problem 1:

Order the following functions so that $f_i \in O(f_j)$ when $i \leq j$.

1. $f_1(n) = 1.1^n$
2. $f_2(n) = \log_2(n) * \log_5(n^9)$
3. $f_3(n) = 9999999999$
4. $f_4(n) = n * 83970^{\log_2(n)}$
5. $f_5(n) = \sqrt{n} + n$
6. $f_6(n) = 666^{\sqrt{n}}$
7. $f_1(n) = n^{\frac{2}{3}}$

Problem 2:

1. Prove that if $f(n) \leq O(g(n))$ and $g(n) \leq O(h(n))$, then, $f(n) \leq O(h(n))$
2. Give the time complexity of the following code in $\Theta(\cdot)$ notation.

Algorithm 1 While Loop

```
$i \leftarrow n$
while $i > 1$ do
    $j = i$
    while $j < n$ do
        $k \leftarrow 0$
        while $k < n$ do
            $k = k + 2$
        end while
        $j \leftarrow j * 2$
    end while
    $i \leftarrow i/2$
end while
```

Problem 3:

There is a straight country road with houses along it. The goverment wants to place some phone base stations at certain points along the road, so that every house is within four miles of one of the base stations. Give an efficient algorithm to achieve this goal with as few base stations as possible. Prove your answer is correct.

Problem 4:

There are some programs that need to be run on a computer. Each program has a designated start time and finish time and cannot be interrupted once it starts. Programs can run in parallel even if their running time overlaps. You have a 'check' program which, if invoked at a specific time point, can get information of all the programs running on the computer at that time point. The running time of the 'check' program is negligible. Design an efficient algorithm to decide the time points at which the 'check' program is invoked, so that the 'check' program is invoked for as few times as possible and is invoked at least once during the execution of every program. Argue that your algorithm is correct.

Problem 5:

Given a circularly sorted integer array consisting n numbers, find the total number of times the array is rotated. Assume there are no duplicates in the array, and the rotation is in the anti-clockwise direction, after n times rotation, the array can be sorted from smallest to largest.Please design an algorithm done in O(log$n$).

For example, array = [3,4,5,0,1,2], then number of rotation is 3.

Problem 6:

You have been hired to plan the flights and you are going to provide service to n cities. This airline will only fly you East. You need to enable all your passengers to travel from any city to any other city (to the East) with a single flight requires Ω(n2) different routes. Devise a set of routes which requires no passenger have more than a single connection (i.e. must take at most two flights), and requires no more than O(nlogn) routes. Prove that your set of routes satisfies these requirements.

Divide Step. Describe the divide step of your algorithm here, making sure to mention what subproblems are produced. Include the total number of routes created during the divide step.

Combine Step. Describe the combine step of your algorithm here. Include the total number of routes created during this combine step

Algorithm. Describe your whole algorithm here. You do not need to repeat the procedures for divide and combine, you're welcome to simply reference them here.

Correctness. Prove the correctness of your algorithm. Namely, show that the selected routes have the property that one can travel from any city to an eastward city with at most a single connection.

Number of Routes. Express the number of routes your algorithm selects as a recurrence relation. Describe, using the master theorem, how you know that this recurrence is O(nlogn).

Problem 7:

Given a n by n wall where n is of form 2^k where $k \geq 1$. The wall has one socket (of size 1 x 1). Bespread the wall (except the socket) using L shaped planks consisting of three 1 x 1 square. Planks are not allowed to cross. You just need to describe and analyze the algorithm.