CS325H: Analysis of Algorithms, Winter 2020

Practice Problems 1

Asymptotic notions

Problem 1. For each of the following, indicate whether $f = O(g)$, $f = \Omega(g)$ or $f = \Theta(g)$.

(a) $f(n) = 12n - 5$, $g(n) = 1235813n + 2017$.

(b) $f(n) = n \log n$, $g(n) = 0.00000001n$.

(c) $f(n) = n^{2/3}$, $g(n) = 7n^{3/4} + n^{1/10}$.

(d) $f(n) = n^{1.0001}$, $g(n) = n \log n$.

(e) $f(n) = n6^n$, $g(n) = (3^n)^2$.

Problem 2. Prove that $\log(n!) = \Theta(n \log n)$. (Logarithms are based 2)

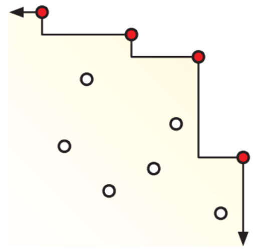
Problem 3. Write a recursive algorithm to print the binary representation of a non-negative integer. Try to make your algorithm as simple as possible. Your input is a non-negative integer n. Your output would be the binary representation of n. For example, on input 5, your program would print '101'.

Problem 4.

(a) Read tree traversal from wikipedia: `https://en.wikipedia.org/wiki/Tree_traversal`, the first section, Types.

(b) Recall that a binary tree is *full* if every non-leaf node has exactly two children. Describe and analyze a recursive algorithm to reconstruct an arbitrary full binary tree, given its preorder and postorder node sequences as input. (Assume all keys are distinct in the binary tree)

Problem 5. Suppose you are given a set P of n points in the plane. A point $p \in P$ is maximal in P if no other point in P is both above and to the right of p. Intuitively, the maximal points define a "staircase" with all the other points of P below it.



A set of ten points, four of which are maximal.

Describe and analyze an algorithm to compute the number of maximal points in P in $O(n \log n)$ time.

Problem 6. Call a sequence $X[1 \cdots n]$ of numbers bitonic if there is an index i with $1 < i < n$, such that the prefix $X[1 \cdots i]$ is increasing and the suffix $X[i \cdots n]$ is decreasing. Describe an $O(\log n)$ time algorithm to search a bitonic sequence of length n for a number k.

More Problems ....

Practice Problem A. Write a recursive algorithm to count the number of binary strings of length n with no consecutive 1's. Your input is a non-negative integer n. Your output should be the number of binary strings of n bits with no consecutive ones. For example, on input 1, your algorithm returns 2 ('1', '0'), on input 2, your algorithm returns 3 ('00', '01', '10'), and on input 3 your algorithm returns ('000', '010', '100', '001', '101').

Practice Problem B. Collatz sequence starting at an integer n is defined as follows. Start with an integer n. In each step, if n is even divide it by two (i.e. $n = n/2$), if it is odd multiply it by three and add 1 to it (i.e. $n = 3n + 1$). Write a "recursive" algorithm to generate Collatz sequence. Can you show that your algorithm ends? What element of recursion is missing here? See `https://en.wikipedia.org/wiki/Collatz_conjecture`.

Practice Problem C. Let $f(n)$ and $g(n)$ be nonnegative functions. Use the definition of Θ-notation to prove that $\max(f(n), g(n)) = \Theta(f(n) + g(n))$.

Practice Problem D. Continuation of Problem 4.

(c) Recall that a binary tree is *full* if every non-leaf node has exactly two children. Describe and analyze a recursive algorithm to reconstruct an arbitrary full binary tree, given its preorder and postorder node sequences as input.

(d) Describe and analyze a recursive algorithm to reconstruct an arbitrary binary tree, given its preorder and inorder node sequences as input.

(e) Describe and analyze a recursive algorithm to reconstruct an arbitrary *binary search tree*, given only its preorder node sequence. Assume all input keys are distinct.

Practice Problem E. Use induction to prove the following facts.

(a) $1 + 2 + \ldots + n = \frac{n(n+1)}{2}$.

(b) $1 + c + c^2 + \ldots + c^n = \frac{c^{n+1}-1}{c-1}$, for any $c > 0$.

Practice Problem F. A shuffle of two strings X and Y is formed by interspersing the characters into a new string, keeping the characters of X and Y in the same order. For example, the string BANANAANANAS is a shuffle of the strings BANANA and ANANAS in several different ways.

$$\texttt{BANANA}_{\texttt{ANANAS}} \qquad \texttt{BAN}_{\texttt{ANA}}\texttt{ANA}_{\texttt{NAS}} \qquad \texttt{B}_{\texttt{AN}}\texttt{AN}_{\texttt{A}}\texttt{A}_{\texttt{NA}}\texttt{NA}_{\texttt{S}}$$

Similarly, the strings PRODGYRNAMAMMIINCG and DYPRONGARMAMMICING are both shuffles of DYNAMIC and PROGRAMMING:

$${}_{\texttt{PRO}}\texttt{D}_{\texttt{G}}\texttt{Y}_{\texttt{R}}\texttt{NAM}_{\texttt{AMMI}}\texttt{I}_{\texttt{N}}\texttt{C}_{\texttt{G}} \qquad \texttt{DY}_{\texttt{PRO}}\texttt{N}_{\texttt{G}}\texttt{A}_{\texttt{R}}\texttt{M}_{\texttt{AMM}}\texttt{IC}_{\texttt{ING}}$$

Given three strings $A[1..m]$, $B[1..n]$, and $C[1..m+n]$, describe an algorithm to determine whether C is a shuffle of A and B. Prove your algorithm is correct and analyze its running time.