CSCI 2300: Introduction to Algorithms

Homework 1

Lucien Brule
Prof. Bulent Yener
April 27, 2023

1 Problem 1

(a) $f = \theta(g)$

(b) $f = O(g)$

(c) $f = \theta(g)$

(d) $f = \theta(g)$

(e) $f = \theta(g)$

(f) $f = \theta(g)$

(g) $f = \Omega(g)$

(h) $f = \Omega(g)$

(i) $f = \Omega(g)$

(j) $f = \Omega(g)$

2 Problem 2

Consider the following psuedocode which takes the integer $n >= n >= 0$ as input:

```
def bar(n):
    print("*")
    if n == 0:
          return
    for i in range(0,n-1):
          bar(i)
```

Let T(n) be the number of times the character "*" is printed by the above code with input $n >= 0$. What is T(n) exactly, in terms of only n ? (ie: not values like T(n-1) or T(n-2)). Prove your answer.

Answer: T(n) $= n + \sum_{i=0}^{n-1} T(i)$

Proof: **Base case:** Let's first verify the base case, n = 0.

When n = 0, the code directly prints "*", and since no recursive calls are made, T(0) = 1.

Now, let's check the equation:

- T(0) $= 0 + \sum_{i=0}^{0-1} T(i)$
- T(0) = 0

The equation doesn't hold true for the base case.

Let's modify the equation for T(n) considering the base case: T(n) = $1 + \sum_{i=0}^{n-1} T(i)$

Now let's verify the base case again: T(0) $= 1 + \sum_{i=0}^{0-1} T(i)$ T(0) = 1 The modified equation holds true for the base case.

Inductive step: Let's assume the modified equation holds true for n = k, and we will show that it also holds true for n = k + 1. We have the following equation for n = k: T(k) $= 1 + \sum_{i=0}^{k-1} T(i)$

Now, let's find the equation for n = k + 1: T(k + 1) $= 1 + \sum_{i=0}^{k} T(i)$ We can rewrite the sum as: T(k + 1) $= 1 + \sum_{i=0}^{k-1} T(i) + T(k)$

From our assumption, we know that: T(k) $= 1 + \sum_{i=0}^{k-1} T(i)$

Substituting this into the equation for T(k + 1), we get: T(k + 1) = $1 + (1 + \sum_{i=0}^{k-1} T(i)) + T(k)$

Simplifying, we get: T(k + 1) $= 1 + \sum_{i=0}^{k} T(i)$

Therefore, the modified equation holds true for $n = k+1$, and by induction, it holds true for all $n >= 0$.

Thus, the correct answer for T(n) is: T(n) $= 1 + \sum_{i=0}^{n-1} T(i)$

3 Problem 3

Problem: Let f(n) and g(n) be asymptotically nonnegative functions. Using the basic definition of θ-notation, prove that $\max(f(n), g(n)) = \theta(f(n) + g(n))$.

Solution: To show that $\max(f(n), g(n)) = \Theta(f(n) + g(n))$, we need to prove that there exist constants $c_1, c_2 > 0$ and $n_0 \geq 0$ such that for all $n \geq n_0$:

$$c_1(f(n) + g(n)) \leq \max(f(n), g(n)) \leq c_2(f(n) + g(n))$$

Lower Bound: Let $c_1 = \frac{1}{2}$. Then, for any $n \geq n_0$ (with $n_0 \geq 0$), we have:

$$c_1(f(n) + g(n)) = \frac{1}{2}(f(n) + g(n))$$

Since $f(n)$ and $g(n)$ are asymptotically nonnegative functions, at least one of them is greater than or equal to half of their sum. Therefore, we can conclude that:

$$\frac{1}{2}(f(n) + g(n)) \leq \max(f(n), g(n))$$

Upper Bound: Let $c_2 = 1$. Then, for any $n \geq n_0$ (with $n_0 \geq 0$), we have:

$$c_2(f(n) + g(n)) = f(n) + g(n)$$

Clearly, the sum of $f(n)$ and $g(n)$ is always greater than or equal to the maximum of the two. Therefore, we can conclude that:

$$\max(f(n), g(n)) \leq f(n) + g(n)$$

Since we have established both the lower and upper bounds, we can conclude that:

$$\max(f(n), g(n)) = \Theta(f(n) + g(n))$$

4 Problem 4

Problem:

Is $2^{2n} = O(2^n$, Why?

Solution:

This statement is false. 2^{2n} is not $O(2^n)$.

To prove this, we need to show that there do not exist constants $c > 0$ and $n_0 \geq 0$ such that for all $n \geq n_0$:

$$2^{2n} \leq c \cdot 2^n$$

Let's assume there exists such a constant $c > 0$. Then:

$$2^{2n} \leq c \cdot 2^n$$

Dividing both sides by 2^n, we get:

$$2^n \leq c$$

However, this inequality is not true for all $n \geq n_0$, because as n approaches infinity, 2^n will also approach infinity, which contradicts the assumption that there exists a constant $c > 0$ that satisfies this inequality.

Therefore, the statement $2^{2n} = O(2^n)$ is incorrect.