

Cosc 30: Discrete Mathematics

Prishita Dharampal

Problem 1. Asymptotic analysis. We often use big-O notation to describe the asymptotic behavior of algorithms. Formally, the proposition $f(n) = O(g(n))$ means:

$$\exists C > 0 \exists N \forall n \geq N, \quad f(n) \leq C \cdot g(n).$$

Prove the following statements (as formal quantified propositions; hand-waving arguments based on intuition of big-O notation are insufficient):

(a)

$$\sum_{k=1}^n k^{30} = O(n^{31}).$$

(b)

$$\sum_{k=1}^n \frac{1}{k} \neq O(1),$$

formally interpreted as

$$\neg \left(\sum_{k=1}^n \frac{1}{k} = O(1) \right).$$

Problem 2. No all-purpose lossless compression. The purpose of this problem is to demonstrate that, even with the materials learned so far, we can already prove a non-obvious fact in computer science.

(a) Write the following statement as a formal quantified proposition:

Every lossless compression algorithm that shortens some input text must lengthen some other input text.

[Hint: It is not sufficient to treat “lossless compression algorithm” as a basic type, for the purpose of the next subquestion.]

(b) Prove the above statement about lossless compression algorithms.

[Hint: The problem may look daunting at first because we have not seen many CS facts proved formally before. Trust the process and carry out the mechanical part of the proof; an insight should strike. Read the first hint if necessary.]

Problem 3. Forward Tower of Hanoi. Consider the following variant of the Tower of Hanoi: the pegs are labelled 0, 1, and 2, and the goal is still to move a stack of n disks (sorted from smallest to largest) from peg 0 to peg 1, such that at any point no disk can sit on top of another disk of smaller size. However, there is one extra constraint: every disk at peg i can only be moved to the next peg $i + 1 \pmod{3}$. In other words, if we imagine the pegs are placed in cyclic order, then every disk can only move forward one step at a time.

- (a) Describe a process to solve this variant of the Tower of Hanoi puzzle. Your algorithm may be described in English or in pseudocode, but explanations are required.
- (b) Prove that your process correctly solves the puzzle in at most $4^n - 1$ steps.
- ★(c) The bound in part (b) can be improved with a better algorithm. What is the fastest method you can come up with to move a stack of n disks from peg 0 to peg 1?

[Hint: If you insist on the optimal bound, you need some combinatorics. Read Pigeon 6.4.3 for inspiration.]

Problem 4*. Approximating square root. On computers we do not usually represent square roots exactly (as solutions to quadratic equations), but as estimations with variable accuracy.

The following procedure `HERON(x)` computes estimations of the square root of x iteratively:

```
HERON(x) :  
  let y0 be an arbitrary real number; set i ← 0  
  while yi^2 is too far away from x:  
    yi+1 ← (yi + x/yi)/2  
    i ← i + 1  
  return yi
```

Heron's method is known at least by the first-century Egyptians and was named after the Greek mathematician Heron of Alexandria. The method converges quadratically (the number of correct digits doubles at each step) and is very efficient in practice. The

goal of this exercise is to prove the correctness of Heron's method.

- (a) State the convergence criteria of $\text{HERON}(x)$ as formal quantified propositions.

[Hint: How do you know the procedure will eventually stop? There are at least two things to check.]

- (b) Prove the convergence criteria you gave in the previous subquestion.