CSE 60111: Complexity and Algorithms
Homework 2

Reminder: For any algorithms question (unless otherwise specified), you must provide the following:

- an algorithm, meaning clear pseudocode, as well as perhaps a brief explanation of the pseudocode if you believe it helpful to clarify the code;
- an analysis of the runtime and space complexity of your algorithm;
- and a proof of correctness for your algorithm.

Each component is required and graded, so be sure you include all of them!

---

1. The residents of the last city on Earth, Zion, must defend themselves from an onslaught of killer flying robots. (Yes, you may have seen this movie.). While they traditionally utilize heavy artillery and kung fu, their newest and most effective method of defense is setting off an EMP to disable incoming robots. They must design an efficient and optimal algorithm to decide when the EMP goes off, so as to kill as many robots as possible as they come in the front gate; however, this is made more complex by the fact that their EMP must be charged, and it gets stronger the longer it charges.

   We formalize this in the following way:

   - Every second (from 1 to n), some number of robots $X[i]$ arrives at the gate, where they can be reached by the EMP if they set it off. Since they have advanced remote sensing techniques, they know all n values in advance.
   - The EMP's charge is described by a function; if it's been charging for k seconds, it's capable of killing $Charge[k]$ or $X[k]$ robots, whichever is smaller (since you can only kill as many as are actually arriving). We'll assume the EMP begins completely drained, so that if it goes off for the first time at second i in the attack, it will kill $\min\{Charge[i], X[i]\}$ robots; from then on, if it was set off at time i and then is next set off at time j, it kills $\min\{Charge[j-i], X[j]\}$ robots.

   Your goal is to design an algorithm that, given the arrivals $X[1..n]$ and the function $Charge[1..n]$, chooses the times to fire the EMP which kill as many robots as possible. (Be sure to justify correctness as well as runtime and space!)

   Example: Suppose $n = 4$, and we have $X = [1, 10, 10, 1]$ and $Charge = [1, 2, 4, 8]$ Then the best solution is to activate at times 3 and 4; at time 3, it will kill $\min 2^{3-1}, 10 = 4$ robots, and then in the 4^{th} second it will kill 1 more robot, for a total of 4 robots. (You can check that any other solution will not kill as many.)

2. DNA strands are sequences of repeated ACGT characters, and they serve as the fundamental instruction manual for building, operating, and maintaining living organisms. It encodes the genetic information necessary for producing proteins, which perform critical cellular functions, and ensures traits are passed down through generations. It also enables growth, reproduction, and cell repair. Viral infections can mutate DNA sequences, by inserting genetic

material into a host cell's DNA process. In this problem, we'll explore variations on sequence matching/alignment that models this process.

(a) Consider a model of virus infection where a virus infects a bacterium, and modifies a replication process by inserting:

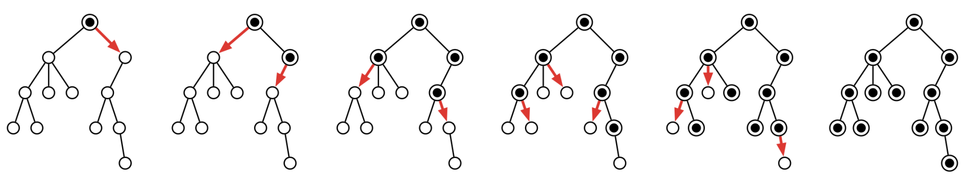
- at every A, between 1 and 5 additional A's
- at every C, a run of 1 to 10 additional C's
- at every G, a run of of G's of arbitrary length ≥ 1
- at every T, a run of of T's of arbitrary length ≥ 1

The gaps or insertions are allowed for in this virally modified final DNA sequence. For example, the sequence AAATTAAAGGGGCCCCCTTTTTTTTTCC is an infected version of ATAGCTC; however, AAAAAAAATTAAAGCCCCCTTTTTTTTTCC would not be, since it inserts too many A's in the first slot and did not insert any extra G's.

Given two sequences $V[1..n]$ and $W[1..n]$, give an efficient algorithm (including run time and space) that will determine if V could be an infected version of W.

(b) Now consider a version where the virus will either delete a letter or will insert a run of arbitrary length, for each A,G,T,C it encounters in the original DNA. Give an efficient algorithm to decide if V could be an infected version of W under these circumstances.

3. Suppose we need to broadcast a message to all the nodes in a rooted tree. Initially, only the root node knows the message. In a single round, any node that knows the message can forward it to at most one of its children. See example below of a message being distributed through a tree in 5 rounds:



Design an algorithm to compute the minimum number of rounds required to broadcast the message to all nodes in an arbitrary rooted tree.