CS 4410: Operating Systems
Homework 5

Instructions for Homework 5:

- Homework may be done in pairs, or individually. If doing in pairs, **one** of you should upload to gradescope and add your partner to the group assignment in the upper right corner of the screen. (Do **not** just upload the assignment twice or it will be graded twice, which means grading will take longer.)
- The deadline is Tuesday, Nov 1st at 11:59AM.
- No late submissions will be accepted.
- You must attribute every source used to complete this homework.
- For some of the problems, you will need two integers. Here is the algorithm for computing these integers:
  - If you are working with a partner, let `var` be the lexicographically smaller of the two NetIDs.
  - Let `varInt` be the integral part of `var`. That is, if `var = rst12`, then `varInt = 12`.
  - If `varInt` is a single digit integer, let `varInt = 13 × varInt`
  - Let `Int1` be the first digit of `varInt`
  - Let `Int2` be the second digit of `varInt`
- **For all problems that use `Int1` or `Int2`, please write down the parameters (related variables and settings calculated from your NetID) before answering each question.**

1 Counting Sheep

Have you experienced Insomnia? Ted once had it when he was thinking about a particular person so that he couldn't fall asleep easily. But luckily (or probably not), he came up with an idea to help him: rather than counting sheep, he decided to simulate a distance vector algorithm in his mind, so that he could soon get bored and sleepy. Actually, interesting though, for some network settings, simulating the algorithm is just like counting sheep. As a TA, he wants to have you do the same as he did in order to have a better understanding of routing algorithms. But this time, hopefully, you don't need to think about it before you sleep. The distance vector algorithm is slightly different from what you have learned from the lecture, but its core is basically the same:

Algorithm 1 Distance vector algorithm

```
▷ D(u,t): cost of u's shortest path to t
▷ D_v(u,t): cost of u's shortest path to t via v
▷ c(u,v): link cost from u to v (subject to changes outside the algorithm)
function INIT(u)
    for all other nodes v do
        if v is a neighbor of u then
            D_v(u,v) = c(u,v)
        else
            D_v(u,v) = ∞
        end if
        D(u,v) = D_v(u,v)
        send D(u,v) to all neighbors
    end for
end function
function LOOP(u)
    while true do
        wait until c(u,v) was changed by d
            or received an update of D(v,t) from v
        if c(u,v) was changed by d then
            for all destinations t that go through v do
                D_v(u,t) = D_v(u,t) + d
            end for
        else if received an update of D(v,t) from v then
            D_v(u,t) = D_v(u,v) + D(v,t)
        end if
        update D(u,t) using D_v(u,t) if it is better than the current one
        if the distance to destination t has changed then
            send D(u,t) to all neighbors
        end if
    end while
end function
```

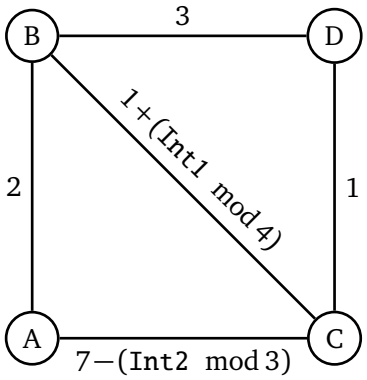
For each node u, the algorithm starts by running `Init`(u) followed by `Loop`(u). Assume the messages (or updates) sent by the nodes are queued, there is no loss, all the nodes have completed `Init` before any of them starts `Loop`, and no node moves to the next iteration of its loop before all of them finishes the same iteration.

1. For the following networking setting (each circle represents a node and each line is the link between two nodes, with the cost on it), write down a table showing $D_v(u,t)$ for each node u after:

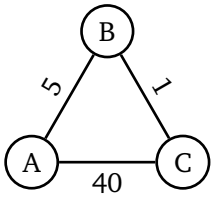
   (a) `Init` is completed

   (b) Each iteration of `Loop` until stable

   The element at ith row and jth column of table u is $D_j(u,i)$.

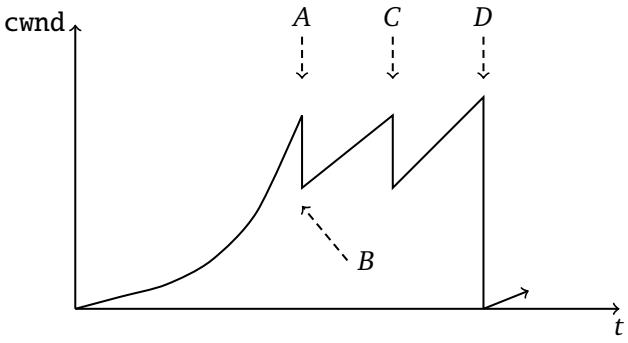


2. Suppose all $D_v(u,t)$ have converged for the following graph. After that, the cost of the link between A and B is changed to $10\cdot(\texttt{Int1}+5)$. What will happen after such change according to the algorithm? Write down the $D_u(x,y)$ for the first five iterations after the link is changed.



2 Secret Intelligence Service

As "M", the head of MI6, you just received an intel from James. After some research, you found that it is a TCP congestion window size graph from the enemy's internal network system, as shown below. To make Jame's effort worthwhile, you decided to fully utilize the information hidden behind the graph. The graph is just a skecth done by James with his cover intact, which means it is not drawn according the scale. The y-axis stands for the TCP congestion windows size of the sender.



- The number of written packet: $N = 2^{4+(\texttt{Int1} \bmod 3)} - 1$,
- Window size at point D: $S = 2^{4+(\texttt{Int2} \bmod 3)}$ packets,

1. The congestion window size of the sender decreases at A, C, and D, when a packet is lost. How does the sender identify that a packet loss occurred at each of A, C, and D?
2. If the event at A happens 1 second after the sender starts the transfer, and the sender has sent N packets to the network before that, what is the RTT of the network? Assume at $t = 0$, the sender tries to open the connection, and also suppose that it can "send" a full window of data instantly, so that the only latency to consider is the propagation delay.
3. Following the previous question, what's the congestion window size of the sender at event B (in number of packets)?
4. Following the previous question, if event C happens 2 seconds after event B, what is the window size at C (in number of packets)?
5. Suppose the window size at event D is S, when it drops to 1. How long will it take for the sender to get back to the window size of S (supposing there is no future packet loss)? Write down your answer in terms of the number of RTTs.

3 Attack from Root

One day, Ted's friend and also the co-founder of CSMA Co., Ltd., Harold, found his computer networking went into a very bad situation. As a genius hacker, he quickly located the problem which stemmed from an unexpected send buffer overflow in his kernel. Almost at the same time, he got a new IRC message on his screen saying "Greetings from Root". Apparently, a hacker named Root has exploited some design flaws in his network subsystem and initiated the attack. As a genius, Harold then realized the problem so he asked you, his new apprentice to help him fix it, because he is very busy on other secret business. Unfortunately, as an employee knows almost nothing about security, you didn't even know the exact principle of the attack, not to mention fixing the bug. So you knocked on Ted's office door and got some hints from him:

Suppose a TCP receiver does not exactly obey what protocol expects it to do, but modifies its own TCP implementation such that, when receiving an arrived TCP packet of N bytes, rather than acknowledge N bytes as a whole, it acknowledges several parts of it by dividing it into M pieces evenly (suppose N is divisible by M) and sending an acknowledge packet for each piece. For example: when it receives data of $N = 800$ and sequence number for the first byte is 1, then, for $M = 2$, it sends 2 acknowledge packets with sequence number 401 and 801 respectively.

1. Suppose that initially during the slow start phase, the congestion window size is 1, the sender sends $N = ((\texttt{Int1} \mod 5) + 1) \cdot 900$ byte packet with the sequence number starting from 1 and the receiver sends $M = 3$ ACKs. What packets will the sender send after receiving 3 ACKs? Specify the sequence number for each packet.

2. Suppose that initially during the slow start phase, the congestion window size is 1, there is no packet loss, transmission time is negligible, the sender keeps sending packets and the receiver keeps using its own malicious TCP implementation. Write down the expression for the congestion window size of the sender after the slow start phase completes, in terms of n (number of RTTs) and M.

3. Is there any simple protocol enhancement to prevent Root from conducting such attack successfully?