22C: 166 Distributed Systems and Algorithms

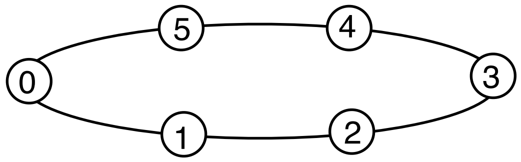
Homework 1, Total points = 60

(Worth 6% of your total grade)

Assigned 2/5/15 due 2/12/15 in class

Please submit typewritten solutions unless you are proud of your handwriting.
Except Q2, only outlines of the algorithms are adequate.

Question 1 (20 points): Consider a ring of 6 processes 0-5 as shown in the diagram below.



Each process i has a non-negative integer variable $p(i)$ that is initialized to 0. Let $N(i)$ represent the neighbors of process i. Each process runs the following program:

{Program for process i } **do** $\forall j \in N(i) : p(i) < p(j)+1 \rightarrow p(i) := p(i)+1$ **od**

The system is asynchronous, i.e. processes execute action anytime after their guards become true.

During an infinite execution, what is the maximum possible difference between the values of p of two different processes? Starting from the initial state, present the steps (i.e. list the sequence of processes that executed the aforesaid action) that via which the difference between two p's can reach the maximum value.

Question 2 (20 points): In 1978, *Tony Hoare* introduced the language CSP (Communicating Sequential Processes) that had a profound influence on the early days of distributed algorithm design (According to Citeseer, his book on CSP was the third most cited work in Computer Science till 2006). In CSP, processes communicate using synchronous message passing (i.e. handshaking): *the sender cannot complete the send operation unless the receiver is ready to receive it, and vice versa*. CSP used the symbols $!$ and $?$ to denote output and input operations respectively. Consider a pair of communicating processes **P** and **Q**. To send the value of a variable v to **Q**, the sending process P uses the instruction $Q!v$, and to receive this value and assign it to a local variable x, process Q uses the instruction $P?x$. Here is an example where a process COPY copies a stream of values generated by process **west** and delivers them to a process **east,** which stores those values

```
program west
define $v$ : integer
do $true \rightarrow COPY!v$ od
```

```
program COPY
define $w$ : integer
do $west?w \rightarrow east!w$ od
```

```
program east
define $x$ : integer
do $COPY?x \rightarrow store\ x$ od
```

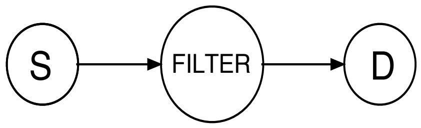


Figure 1. FILTER will allow only "useful data" generated by S to reach to D.

Now, consider a system of three processes S, FILTER, D as shown in Figure 1. Process S generates an arbitrary data stream. Process FILTER is meant to allow only “useful data” generated by S to reach to D: it will only allow values between 5 and 10 to reach D, and will discard all duplicate values. Assuming that each process has an unlimited amount of data storage, write a program in CSP for each of the three processes (*Feel free to read the original paper on CSP*).

Question 3 (10+10=20 points): Consider a population of size n in a town, and the task of multicasting a message m to every resident of that town. One resident starts the multicast, and residents communicate using Twitter. Communication is bi-directional (i.e. the communication graph is undirected) and round-based: *in each round, a person can tweet a message to exactly two other persons*, and every person receiving the tweet can forward it to *two other persons* simultaneously. The goal is to complete the multicast as quickly as possible.

Part 1. Assume that each sender has full knowledge of those who have not yet received the message. Also assume that no two persons will tweet to the same non-recipient in any round. Propose an algorithm (outline only) using which the broadcast is completed in the *fewest number of rounds* (only the main idea using pseudo-codes is needed here). Calculate the time complexity in rounds.

Part 2. Now assume that senders have no knowledge of who has already received the message (senders are lazy and no one maintains the list of the residents to whom s/he has already tweeted the message in the previous rounds). So each sender randomly picks two residents, and tweets them (unless there is only one resident left, in which case the tweet will be directed to that resident only). As a result, a resident who has already received the tweet can receive another tweet that does not contribute to the progress of the multicast. Also, a resident who has not yet received the tweet, can receive a tweet from multiple senders, which may slow down the progress.

(a) Run a simulation experiment to compute the expected number of rounds $E(n)$ needed for the message to reach every resident. Use 10 different values of n ranging from 100 to 10,000, and in each case compute $E(n)$ by repeating each experiment 50 times. Show the graph $E(n)$ vs. n, and try to empirically guess the time complexity in rounds.

(b) Can the residents figure out when the algorithm will terminate?