Tutorial 8 (CSC-503)

1. Write a program in C++ / Java to implement BSS protocol. Your program should take as input a description of several process schedules (i.e., lists of send and recv_B operations). Each send is a broadcast. Each site runs a process implementing BSS protocol. All messages are first received by BSS process. BSS process delivers them to the application. For each process the output of your program should show send, recv_B and recv_A operations annotated with vector clock of the corresponding process at that time.

The input of the program will be a collection of processes, each with a list of send and recv_B operations. The processes are named p1...pn for some n (you may assume that n is at most 9) The format of a process in input is:

```
begin process p1
operation
…
operation
end process p1
```

where each line contains a send or receive operation.

- send msg (that is, broadcast message msg to all processes)
- recv_B pN msg (that is, BSS receives message “msg” from process pN)

where msg is any alphanumeric string.

The send operation simply broadcasts a message and the process is free to continue executing the next operation. The recv_B operation indicates that the message has been received by BSS process. For each recv_B operation output should show an additional recv_A operation which shows when the msg is delivered to the application. The order of events “send” and “recv_B” in the output should be same as in the input given to the program, and a message is always sent before it is received.

Here is a small example illustrating the input format:

```
begin process p1
send m1
end process

begin process p2
recv_B p1 m1
send m2
end process p2

begin process p3
recv_B p2 m2
recv_B p1 m1
end process p3
```

Note that the application running on process 3 should receive message p1 first and then p2.

The output should show these processes with additional recv_A operations. For each recv_B operation (event) in the input there will be a recv_B and a corresponding recv_A operation in the output. Additionally, each operation in the output should show the vector clock value of the process at the time of the operation (event). Operations in the output are:

● send msg T (that is, message msg was sent at local time T)
● rece_A pN msg T (that is, message msg is received by application from pN at local time T)
● rece_B pN msg T (that is, message msg is received by BSS from pN at local time T)

The following is a valid output:

```
begin process p1
send m1 (100)
end process

begin process p2
recv_B p1 m1 (000)
recv_A p1 m1 (100)
send m2 (110)
end process p2

begin process p3
recv_B p2 m2 (000)
recv_B p1 m1 (000)
recv_A p1 m1 (100)
recv_A p2 m2 (110)
end process p3
```

Submit the following:

1. Your program (name of the file should be myBSSProg_enrollno i.e. if your enroll no is 2111101 name of the file should be myBSSProg_2111101)
2. A README file stating that how your program should be complied and executed. Additionally it should also have sample input (on which you have tested your program) and the output for the sample input

   Submit a zip / rar file (its name should be your enrollment no) containing files myBSSProg_enroll_no and README

Marks: Your program runs on your input – 35
Your program runs on our correct input (different from above) – 35
Your program detects our incorrect input and gives error message – 10
Your understanding about your program - 20