**HARIKRISHNAN GOPAL JANAKIRAMAN**

**PROJECT -3 (PRODUCER AND CONSUMER)**

[**HARIKRISHNAN\_GOPALJANAKIRAMAN@STUDENT.UML.EDU**](mailto:HARIKRISHNAN_GOPALJANAKIRAMAN@STUDENT.UML.EDU)

It is a producer-consumer problem requiring an arbitrary number of producers (of donuts), an arbitrary number of consumers and a buffer manager. The selection of the ring buffers are done by a Random Generator .A node controller is present on each host which could have either the producer or consumer processes running on it.

The buffer manager provides and controls storage for 4 ring buffers and their required control variables. Initially enters into an endless loop waiting for connection requests from a consumer or producer which it handles these requests in child processes.

The node controller run on 3 hosts and are implemented by hosts using Ricart Agarwal's

algorithm. The threads will be listening on the ports. The thd\_startNodeController

waits for two nodes get connected. After the connection of two threads,

thd\_serverListen(), producer(), consumer() threads, until node controller started accepting

new message. Timestamping for the messages are done based on Ricart Agarwal's algorithm.

**Included Files:**

node\_control.c

node\_util.c

buffer\_manager.c

buffer\_util.c

**Hosts:**

Buffer Manager : 192.168.8.151

Node Controller 1 : 192.168.8.152

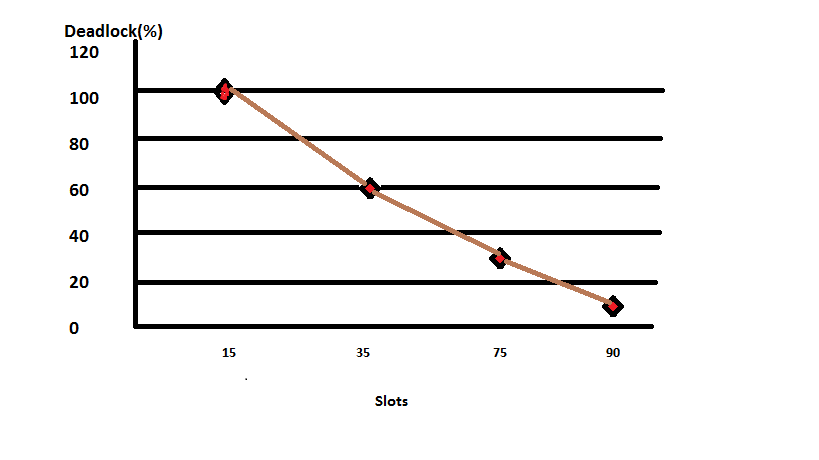
Node Controller 2 : 192.168.8.153

Node Controller 3 : 192.168.8.154

**Test case 1:**

Producers are kept at constant 5 and consumers are kept at constant 15. But the queue depth size

are varied. The observations are plotted as follows.

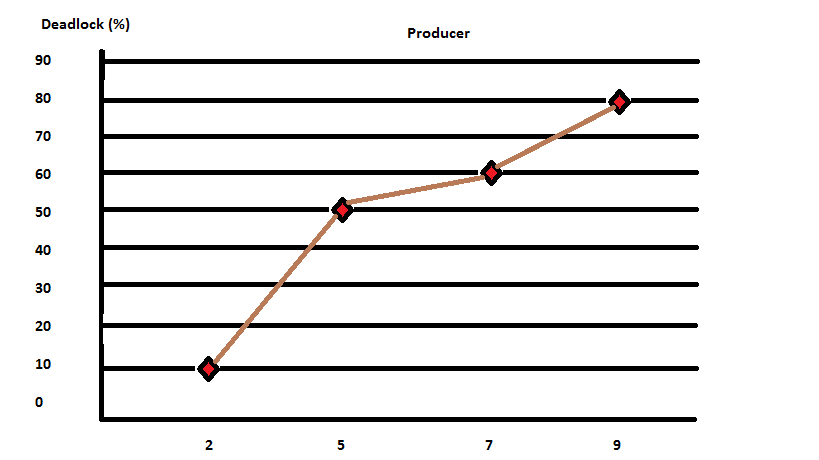


**Test case 2:**

50% Deadlock probability from the previous case is recorded and found to be around 50 slots.

Slots are kept at constant 50 and the producers are varied. The consumers are kept at 5. The

observations are plotted as follows.



**Test case 3:**

Finally, the producers are kept at constant 5. The queue size is maintained at 50. But the

consumers are varied . The output is calculated as follows.

