


National University of Computer and Emerging Sciences, Lahore Campus

	Course Name:	Parallel and Distributing Computing	Course Code:	CS3006
	Degree Program:	BS (CS)	Semester:	Spring 2023
	Exam Duration:	60 Minutes	Total Marks:	35
	Paper Date:	--/04/23	Weight	12.5
	Exam Type:	Mid II Retake	Page(s):	5

Student : Name: _____ Roll No. _____ Section: _____

Question # 1 a: MCQs [6 marks, CLO # 1]

From the given options, select the best answer.

- i. When calculating message passing costs, we would assume that “tw” would be effected by?
 - a. Switch latencies
 - b. Number of hops
 - c. Time needed to add headers
 - d. **Bandwidth of the links**
- ii. For a $2*16$ mesh, using a naïve solution, how many messages would you expect the sending process to send, for a 1-to-all broadcast?
 - a. Less than 22
 - b. **More than 30**
 - c. Less than 10
 - d. Less than 16
- iii. With all-to-1 reduction, we are essentially?
 - a. Requiring all (p-1) processes to send messages to all p process
 - b. Requiring 1 process to send messages to (p-1) processes
 - c. **Requiring all (p-1) processes to send messages to 1 process**
 - d. Requiring all p processes to send messages to 1 process
- iv. Which of the following is not an issue in distributed systems?
 - a. Programming complexity
 - b. **Scalability**
 - c. Scarce robustness
 - d. Hard to optimize
- v. Simple Storage Service (S3) is an example of:
 - a. **Infrastructure as a Service**
 - b. Platform as a Service
 - c. Software as a Service
 - d. None of the given options

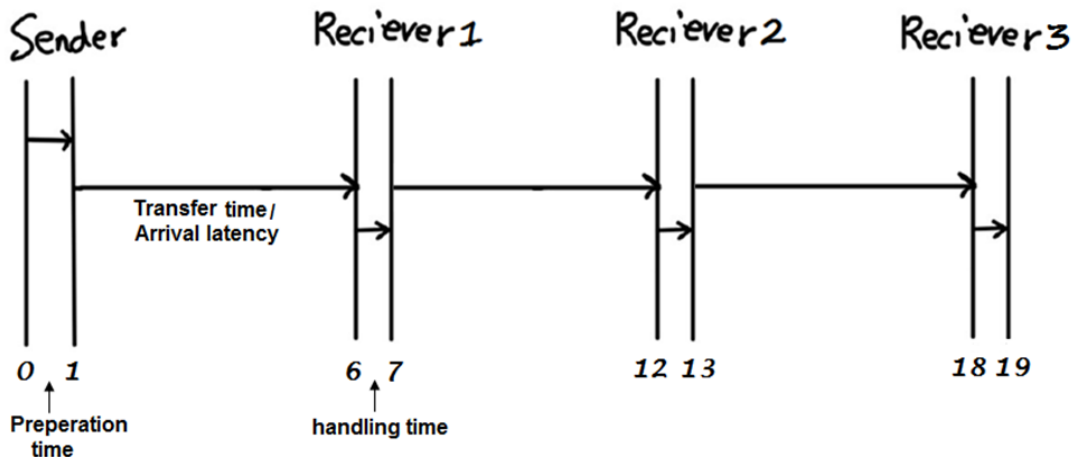
Question # 1 b: [3 marks, CLO # 1]

True/False

- I. Distributed Operating Systems are better suited for heterogeneous multi-computers.
 - a. True
 - b. False
- II. Cluster Computing is a form of Utility Computing.
 - a. True
 - b. False
- III. Grid Computing is biased towards general purpose computing.
 - c. True
 - d. False

Question # 1 c: [6 marks, CLO # 1]

Calculate the time required to transfer 400 mbits of data from Sender to Receiver3. Bandwidth of the link is 10 mbits/s.



$$t = t_s + (t_w \cdot m + t_h + t_r) \cdot n$$

$$t_s = \text{Preparation Time} = t_1 - t_0 = 1 - 0 = 1 \text{ s}$$

$$t_w = 1 \text{ unit transfer time} = \frac{1}{r} = \frac{1}{10} = 0.1$$

$$m = \text{message size} = 400$$

$$t_h = \text{Arrival Latency} = t_2 - t_1 = 6 - 1 = 5$$

$$t_r = \text{Receiver Handling Time} = t_3 - t_2 = 7 - 6 = 1$$

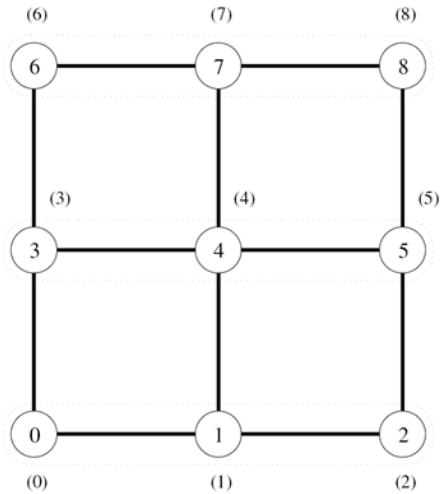
$$n = \# \text{ of hops} = 3$$

$$t = 1 + (0.1(400) + 5 + 1) \cdot 3$$

$$t = 139.5$$

Question # 2: [4+3+3 marks, CLO # 3]

a) Explain the All-reduce communication operation on the following mesh with 9 nodes.



b) Provide total cost estimation for this operation. You have to consider the size of each message!

c) What modification would be required in the following code to perform All-reduce operation?

```
1.  procedure ALL_TO_ALL_BC_MESH(my_id, my_msg, p, result)
2.  begin

    /* Communication along rows */
3.      left := my_id - (my_id mod  $\sqrt{p}$ ) + (my_id - 1) mod  $\sqrt{p}$ ;
4.      right := my_id - (my_id mod  $\sqrt{p}$ ) + (my_id + 1) mod  $\sqrt{p}$ ;
5.      result := my_msg;
6.      msg := result;
7.      for i := 1 to  $\sqrt{p} - 1$  do
8.          send msg to right;
9.          receive msg from left;
10.         result := result  $\cup$  msg;
11.     endfor;

    /* Communication along columns */
12.     up := (my_id -  $\sqrt{p}$ ) mod p;
13.     down := (my_id +  $\sqrt{p}$ ) mod p;
14.     msg := result;
15.     for i := 1 to  $\sqrt{p} - 1$  do
16.         send msg to down;
17.         receive msg from up;
18.         result := result  $\cup$  msg;
19.     endfor;
20. end ALL_TO_ALL_BC_MESH
```

Question # 3: [6+4 marks, CLO # 2]

- a) Write the output for the following piece of code assuming that there are 4 MPI processes. Assume there is no syntax error.

```
#include <mpi.h>
#include <stdio.h>

int main (int argc, char** argv) {
    MPI_Init (NULL, NULL);
    MPI_Status status;

    int p, b, my_rank;
    MPI_Comm_size(MPI_COMM_WORLD, &p);
    MPI_Comm_rank(MPI_COMM_WORLD, &my_rank);

    int a = my_rank+10;
    int sTag = my_rank;
    int rTag = (my_rank - 2 + p) % p;
    int next = (my_rank + 2) % p;
    int prev = ((my_rank - 2 + p) % p);

    MPI_Sendrecv(&a,1,MPI_INT,next,sTag, &b,1,MPI_INT,prev,rTag, MPI_COMM_WORLD, &status);
    printf("I am %d: Got:%d from %d and Sent:%d to %d\n ", my_rank, b, prev, a, next);
    MPI_Finalize();
}
```

- b) Describe one reasonable scenario where we would use MPI_ANY_SOURCE?
- c) When we want all processes to synchronize, which MPI function should we call?