COL380 Practice Questions

1. Suppose the iso-efficiency function for a parallel solution is $\mathcal{I}(p) = \Omega(p^2)$. Given that the solution is known to take $O(n^2)$ steps on a single processor for input of size n, analyze the scaling behavior of the algorithm.

2. What is the summary message of Amdahl's law?

3. One way to maintain sequential consistency among operations is to find an explicit order as they are performed. A distributed log of all operations can help with this. Your task is to append each operation to the end of the log. Use CAS (Compare-and-Swap) to implement this append such that all operations are added safely (in some order).

4. Suggest a message passing based algorithm to maintain the log described in the previous question.

5. Suppose we have an $N \times N$ matrix **M** distributed block-row wise among P MPI processes. Process i has rows $i\frac{N}{P}$ to $(i+1)\frac{N}{P}-1$. The sub-matrices are stored in row-major order. Provide MPI-like code to redistribute **M** such that processor j receives columns $j\frac{N}{P}$ to $(j+1)\frac{N}{P}-1$ stored in column-major order. Pay particular attention to efficiency. Explain your approach.

6. What logic problems are exhibited by the 2-process producer-consumer pseudo-code below? How would you fix those problems? Assume buffer (including endbuffer) is already allocated on both ranks, and they are large enough to hold all messages. Assume some pre-arranged marker is used to check for the end_condition.

```
Rank 0:                                  Rank 1:
  while(more to produce):                  while(true):
      Produce(buffer)                          MPI_Irecv(buffer) from rank 0
      MPI_Isend(buffer) to rank 1              MPI_Barrier
      MPI_Barrier                              if(buffer has END) break
       MPI_Rsend(ENDbuffer)                          Consume(buffer)
```

7. The goal of the following code is to send only the even rows and columns of a Matrix. Does it succeed? Provide the list of the actual ints received in the buffer of the MPI_Recv, (whichever is matched with the MPI_Send on line 9 below).

```
#define N          4
1: int matrix[N][N];
2: MPI_Datatype first, second, final ;
// Fill matrix Data
3: for (int i = 0; i < N; i++)
4:    for (int j = 0; j < N; j++)
5:       matrix[i][j] = i*N + j + 3;
6: MPI_Type_vector(N/2, 1, 2, MPI_INT, &first);
7: MPI_Type_vector(N/2, 1, 2, first, &final);
8: MPI_Type_commit(&final);
9: MPI_Send(matrix,1,final,dest,99,MPI_COMM_WORLD);
// Expect matching.. MPI_Recv(buffer,MAX,MPI_INT,..);
```

8. Recall MPI one-sided memory Read (*Get*) and Write (*Put*) operations. These are like shared memory operations where variables are not addressed by an alphanumeric name but by a <window, rank, offset>

triple. Explain how these operation should be implemented by the owner process of the window receiving *Get* and *Put* requests through MPI_Recv (in a dedicated MPI internal thread), and returning the read data or write acknowledgement after completing the operation locally. What memory consistency model does that imply? Explain.

9. Explain the terms Work, Cost, Efficiency, and Speedup.

10. Suppose the time to sort in parallel $T(n,p) = \frac{(n \log n)}{p} + \log n$. Derive efficiency. Derive iso-efficiency.

11. Does a busy-wait synchronization protocol (i.e. while(cond);) imply absence of lock-freeness? Explain.

12. Does a busy-wait synchronization protocol (i.e. while(cond);) imply absence of wait-freeness? Explain.

13. Does the following code ensure mutual exclusion of P1 and P2 sharing variables sv1 and sv2? Assume sequential consistency. Is progress guaranteed? Is it lock-free? Wait-free?

P0	P1

Exclusive(Func f):
  while (flag0 == flag1) ;
  f();
  flag0 = flag1;
	Exclusive(func f):
	  while !(flag0 == flag1) ;
	  f();
	  flag1 = !flag0;

14. When a Rendezvous envelop arrives at a recipient MPI stub with <dest, tag, com> triple. List all possible actions the stub may need to take, and the conditions under which it would take them.

15. What are the possible actions needed to be taken by the MPI stub in response to an MPI_Recv call in user code? List the actions and the conditions under which each is taken.

16. Suppose a graphs is distributed on multiple MPI ranks. Assume an adjacency list representation of the graph, where each vertex is identified by rank r of the process where its home is and its integer index in a local array on that process. Each vertex is assigned a unique rank as its home. V[i] at rank r maintains a list of <rank, index> tuples, each identifying one of its adjacent vertices. Provide MPI-like code for BFS traversal of this graph.

17. Implement MPI_B*arrier*.

18. Implement distributed Lock using message passing?

19. In two message-passing processes (each with a single thread of execution), implement function TimedCall(uint32 num, Func func, Arg args), which calls func(args) ensuring that calls are made in order of their 'num' parameter. All calls must be made but no call for a lower num may be made after a call for a higher num has been made (on either process). Assume that num for each process increases monotonically. State any additional assumptions.

20. A city has G gates. The guard deployed at each gate can count the number of people entering or leaving the city through that gate. The capacity of the city is N, and at any time if allowing a person in at any Gate I (I < G) will N, entry must be denied. Guards can only gather information by passing messages to each other. Provide a distributed protocol to help each guard decide whether to let the next person in. What are safety and progress properties of your protocol?