COL 380	April 21, 2016
Lab 3	
Instructor: Subodh Sharma	**Due: April 26, 23:55 hrs**

NOTE:
a) This labs is to be performed individually.

1 Random Key Generation

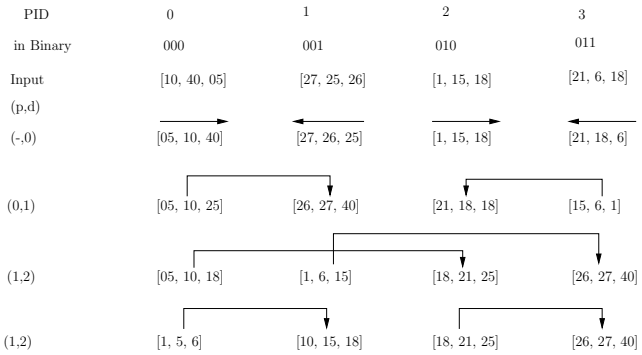
Write an MPI program wherein the root process generates a million unique random keys with values in the range $[1, 100]$ and distributes them to other processors in an efficient manner. Additionally, in a single go the root process can allocate and generate no more than 100,000 keys. Program must output the time taken by the MPI program to perform the afore-mentioned task.

2 Bitonic Sort

Use MPI to implement Batcher's Bitonic Sort by first using only blocking send and receive calls and then overlapping computation and communication using non-blocking send and receive calls. Compare the performance of two versions. Use the result of previous exercise to intialize the computation for this problem.

2.1 Explanation of Bitonic Sort

The overall idea is that each process (total number of processes must be a power of 2) will contain some portion of enteries which the process sorts either in *ascending or descending* order depending on the rank of the process and the progress of the computation. We begin with an example figure shown below to explain the working of the Bitonic sort. The protocol works in the following way:



- (p, d): p is the bit position in which processes with differing values pair-up to perform merging. For instance with (0,1), we observe that processes 0 and 1 pair-up because the bit in position 0 of PID's bit representation are differing. d is the direction bit. The d^{th} bit in the bit representation of PIDs is when 0 the direction of sort is ascending, otherwise it is descending.
- Note that p is never equal to d. When p becomes $d - 1$ then in subsequent phases, the value of p start to decrease.

- $(-, 0)$ indicates sort of process-local entries.
- Arrows indicate the processes merging their local entries.

3 Grading

Your parallel solution would be checked for correctness and performance. Students are expected to give a demo of their work to the TAs along with the submission of a lab report not exceeding 400 words. Lab report must clearly specify design decisions, scalability arguments, communication overhead difference between blocking and non-blocking communication calls and the execution times of programs.

- Correctness: 20 marks
- Performance: 20 marks
- Lab report: 10 marks