Assignment 5
CS4823/CS6643 Parallel Computing

1 Readings

Read Chapters 4, except 4.6 (circular shift), Chapter 6, 6.3-6.6.7,

Chapter 7, 7.1-7.4, 7.10, Load balancing papers posted.

2 Problems

1. (10 points) Embedding binary tree on 2d-mesh: Prove that a binary tree of depth $k > 4$ does not have a load-1, dilation-1, congestion-free embedding on a 2d-mesh. For this, show by induction that $2^{k+1} - 1 > 2k^2 + 2k + 1$ for $K > 4$.

2. Consider the all-reduce operation in which each processor starts with an array of m words, and needs to get the global sum of the respective words in the array at each processor. This operation can be implemented on a ring using one of the following three alternatives:
   (i) All-to-all broadcast of all the arrays followed by a local computation of the sum of the respective elements of the array.
   (ii) Single node accumulation of the elements of the array, followed by a one-to-all broadcast of the result array.
   (iii) An algorithm that uses the pattern of the all-to-all broadcast, but simply adds numbers rather than concatenating messages.

   (a) (9 points) For each of the above cases, compute the run time in terms of m, ts, and tw.

   (b) (1 point) Assume that ts = 100, tw = 1, and m is very large. Which of the three alternatives (among (i), (ii) or (iii)) is better?

   (c) (1 point) Assume that ts = 100, tw = 1, and m is very small (say 1). Which of the three alternatives (among (i), (ii) or (iii)) is better?

3 MPI Programming

1. (20 points) Write a MPI program on Fox server to multiply two n-by-n matrices using p processors with $1 \leq p \leq 8$. Fill up the matrices with some constant values so that it would be easier for you to verify the resulting matrix for correctness. You may use collective operations or send/receive primitives.

2. (5 point) Prepare a speedup plot (Ts/Tp) or a table with varying n and vary number of processes in the available range. Use pure sequential time with three nested loop for Ts.

Hint: A 2D array (matrix) is essentially a 1D array with row-major order. You may implement the sequential code in the same program and time it, followed by parallel code and time that, and calculate the speedup. Experiment and choose sufficiently large n for a reasonable speedup. Larger n may result in better speedup. Po should generate the input.

Submission: Submit your source code and plot/table.