CS484: Parallel Programming - Spring 2017
HW3
Due Date : April 30th, 11:59PM
Submission Method : SVN

April 16, 2017

1. **Isoefficiency**

   Compute isoefficiency functions for the following parallel algorithms. Assume a logP communication model where the cost of a message is $\alpha + \beta * m$ where α is the latency of the message, β is the cost of transmitting a byte and m is the message size in bytes. Assume that a process can send/receive a single message at a time.

   (a) Consider a distributed matrix-vector multiplication of an NXN matrix M with an $NX1$ vector Y on a group of P processes. Assume the matrix and vector are already distributed row-wise, where each process contains $\frac{N}{P}$ rows of the matrix, and the vector is distributed in chunks of $\frac{N}{P}$ elements. The figure below explains the data decomposition.

   The algorithm for sequential matrix-vector multiplication is as follows:

   ```
   for (i=0; i<N; i++)
     for(j=0; j<N; j++)
       Z[i] += M[i][j]*Y[j];
   ```

   Assume M is already distributed *row-wise*, where each process contains $\frac{N}{P}$ rows of M, and Y and Z are distributed in chunks of $\frac{N}{P}$ elements.

   - Assemble the entire Y vector on each process. (You can do this using an all-to-all algorithm on a hypercube [refer to the lecture notes]).
   - Compute the local chunks of Z by multiplying the rows of the matrix with the vector Y. You can leave the vector Z distributed.

   Compute the communication cost and the isoefficiency funstion of this algorithm.

   (b) What is the minimum communication cost and isoefficiency function of a parallel prefix algorithm for a set of N integers distributed across P processes? Assume $N\%P == 0$.

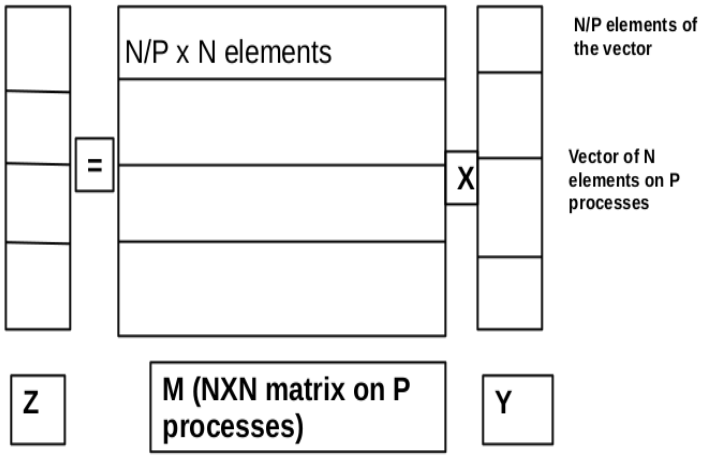

Figure 1:

2. **Topologies**

   (a) Suppose you have a network arranged in a 3D mesh topology with size 30x40x50 on the x, y, and z axes respectively.

      i. What is the diameter and degree of the above topology?

      ii. Assuming a link bandwidth of 4 GB/s, what is the bisection bandwidth of this network? Explain your answer.

      iii. Suppose now the bandwidth along each axis is different: links parallel to the x axis have a bandwidth of 2 GB / s, while the remaining links have a bandwidth of 4 GB /s. What is the bisection bandwidth of this new network?

   (b) Suppose you have to map a 200x300 2D grid of processes onto the above 30x40x50 network; i.e., for each process p, p[i,j] communicates only with p[i-1,j], p[i+1,j], p[i,j-1], and p[i,j+1]. Describe a mapping scheme for assigning processes to nodes; your scheme should assign process[i,j] to node[x,y,z]. State your assumptions if any, and criteria behind choosing a mapping scheme.

3. **Sample Sort**

   You have to implement a parallel sorting algorithm by sampling splitters from the dataset. A stub code is provided for you. Please complete the code provided with the necessary MPI calls.

   - The dataset can be initialized locally (fill my_data with random values). No further communication is necessary for the initialization step. Sort my_data using the STL sorting function.
   - Every process randomly selects some number of sample keys from its local data. The number of samples chosen on each local process is called the oversampling

ratio.The default oversampling ratio is set to 20 in the skeleton code.Samples from all processes should be gathered at the root (process 0).

- Process 0 sequentially sorts the gathered keys (also called probes), using a sequential sorting algorithm. This code has been provided with the skeleton code. Note that the code adds a maxkey at the end of the probes. Process 0 broadcasts the sorted probes to all processors.
- Each processes counts the number of keys it has between consecutive probes. Compute the histogram of counts using a reduction of local counts.
- Process 0 selects $p - 1$ splitter keys from the histogram using the findSplitters function provided in utils.cpp.
- Process 0 broadcasts p splitters (including maxkey). Define a bucket as the range of values that span [splitter[i-1], splitter[i]). Assign bucket i to process i. The processes then exchange non-local data.
- Locally sort all received data using a sequential sorting algorithm. The skeleton code computes initial and final checksum of the data after sorting. If your code runs correctly, the initial and final checksums are likely to match.

Experiments :

- For large problem sizes, vary the oversampling ratio and observe the effect of oversampling on the maximum bucket size.
- For a given problem size, vary the number of processes, plot a graph of the execution time of sample sorting with increasing number of processes.

What to submit: Turn in the entire code. Make sure that your code compiles using the makefile provided. Write a document (preferrably pdf) with the graphs and any additional observations you may have made regarding sample sort.