CS 484: Parallel Programming Spring 2017
Homework 3 (Due April 30th, 11:59PM)

Submitted by:
Kevin Rajan **ksrajan2**

1. **Isoefficiency**

   Compute isoefficiency functions for the following parallel algorithms. Assume a logP communication model where the cost of a message is $\alpha + \beta * \text{m}$ 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, P and the vector is distributed in chunks of $\frac{N}{P}$ elements. The figure below explains P the data decomposition.

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

```
void seq_mat_vec_mul(int* Z, int** M, int* Y, int N) {
    for (int i = 0; i < N; i++) {
      for(int 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.

SOLUTION GOES HERE

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?

$$diameter = (x-1) + (y-1) + (z-1)$$

$$=> 29 + 39 + 49 = 117$$

$$degree = [3, 6] \ (range\ of\ 3\ to\ 6,\ inclusive)$$

The diameter of this given mesh is **117** and the degree for each node is between **3 and 6**.

ii. Assuming a link bandwidth of 4 GB/s, what is the bisection bandwidth of this network? Explain your answer.
If this were a 2D mesh of size 30x50, the bisection width would be **30**. For a 3D mesh of 30x40x50 we just have 40 layers of the 30x50 mesh stacked on each other. Hence, the bisection width of this would be

$$bisection\ width = 30 * 40 => \mathbf{1200}$$

To find the bisection bandwith of this mesh, we need to multiply the bisection width by the channel bandwith which is given as **4 GB/s**.

$$bisection\ bandwith = channel\ bandwith * bisection\ width$$

$$=> 4\ (GB/s) * 1200 = \mathbf{4800\ (GB/s)}$$

Given the link bandwith of 4 GB/s, the resulting bisection bandwith of the network would be **4800 GB/s**.

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?
Since we still want to minimize the overall bandwith usage, we should take advantage of the links that have 2 GB/s channel bandwiths. This limits us to cutting parallel to the x-axis this time as opposed to z-axis as last time.

$$bisection\ width = 40 * 50 => \mathbf{2000}$$

To find the bisection bandwith of this mesh, we need to multiply the bisection width by the channel bandwith of the links we cut which is given as **2 GB/s**.

$$bisection\ bandwith = channel\ bandwith * bisection\ width$$

$$=> 2\ (GB/s) * 2000 = \mathbf{4000\ (GB/s)}$$

(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.

Assumptions:

- The original 2D grid of processes are labeled with i, j and k where i ϵ [0, 199,] j ϵ [0, 299] and k ϵ [0, 0]
- The 3D grid of processes defined above are labeled with i′, j′, k′ where i′ ϵ [0, 29], j′ ϵ [0, 39], k′ ϵ [0, 49]
- The total number of nodes in each case is the same:

$$(200 * 300 == 60000 == 30 * 40 * 50)\ is\ true$$

There are several mappings that can be used in this situation. However, I believe that the simplest mapping would take advantage of the following facts:

$$30 * 10 = 300$$

$$\frac{40 * 50}{10} = 200$$

Using these two facts, the following mapping can be created simply.

$$f(i', j', k') = (j/10, i/5, i/4)$$

This mapping maps the y-axis from the original grid to the x-axis of the 3D mesh and uses the x-axis from the original grid to create the y and z axes of the 3D mesh.

3. **Sample Sort**