High Performance Computing : Assignment 1

Oral questions

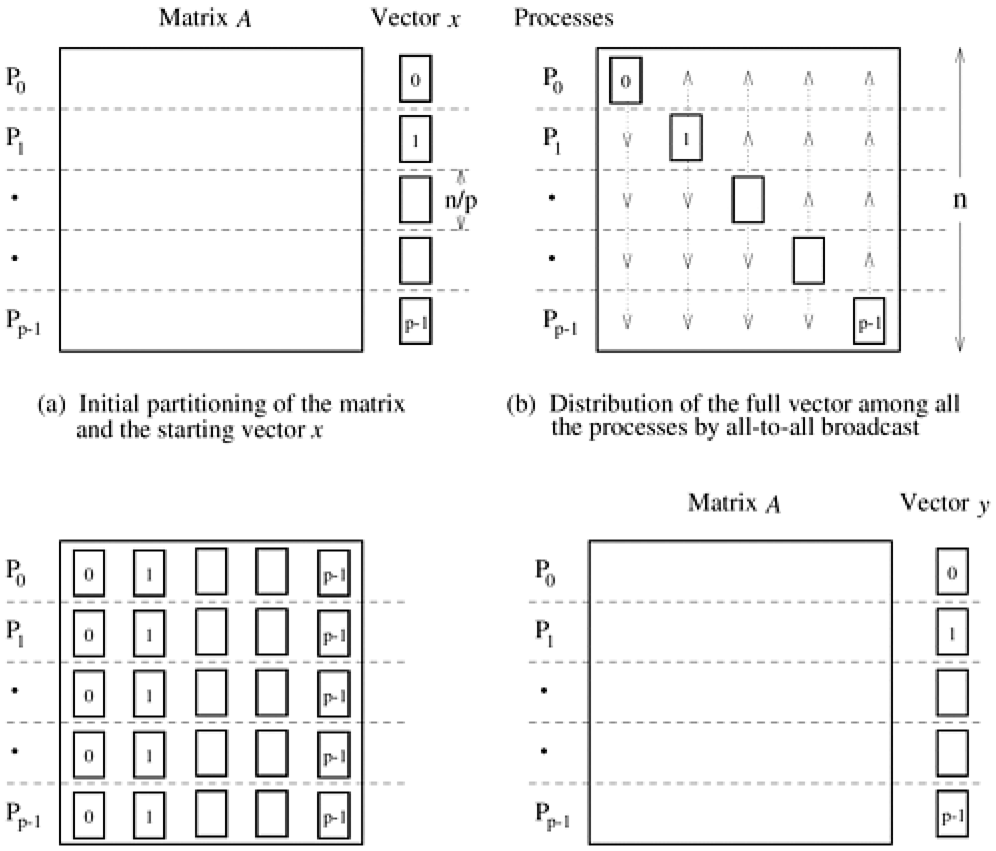
1. What are the three distinct parallel formulation of matrix vector multiplication.

   Row-wise 1D partitioning , Column-wise 1D partitioning , 2-D partitionig

2. Explain rowwise 1-D partitioning

One Row Per Process

First, consider the case in which the *n* x *n* matrix is partitioned among *n* processes so that each process stores one complete row of the matrix. The *n* x 1 vector *x* is distributed such that each process owns one of its elements.



(a) Initial partitioning of the matrix and the starting vector *x*

(b) Distribution of the full vector among all the processes by all-to-all broadcast

(c) Entire vector distributed to each process after the broadcast

(d) Final distribution of the matrix and the result vector *y*

3. Explain columnwise 1-D partitioning.

   Same as row just column instead

4. What is parallel runtime of matrix vector multiplication

$$T_P = \frac{n^2}{p} + t_s \log p + t_w n.$$

5. Explain 2-D partitioning.

One Element Per Process

We start with the simple case in which an *n* x *n* matrix is partitioned among n^2 processes such that each process owns a single element. The *n* x 1 vector *x* is distributed only in the last column of *n* processes, each of which owns one element of the vector. Since the algorithm multiplies the elements of the vector *x* with the corresponding elements in each row of the matrix, the vector must be distributed such that the *i*th element of the vector is available to the *i*th element of each row of the matrix.

6. What is parallel run time of one row per process using row wise 1-D partitioning.

$$T_P = \frac{n^2}{p} + t_s \log p + t_w n.$$

7. What is parallel run time of one row per process using row wise 1-D partitioning.

8. Compare 1-D and 2-D partitioning

A comparison of Equations 8.2 and 8.7 shows that matrix-vector multiplication is faster with block 2-D partitioning of the matrix than with block 1-D partitioning for the same number of processes. If the number of processes is greater than *n*, then the 1-D partitioning cannot be used. However, even if the number of processes is less than or equal to *n*, the analysis in this section suggests that 2-D partitioning is preferable.

Among the two partitioning schemes, 2-D partitioning has a better (smaller) asymptotic isoefficiency function. Thus, matrix-vector multiplication is more scalable with 2-D partitioning; that is, it can deliver the same efficiency on more processes with 2-D partitioning than with 1-D partitioning.

9. What is parallel runtime of matrix vector multiplication?

repeat

10 Is it possible to implement parallel matrix vector multiplication using a cluster environment?

Practice problem

- Implement parallel matrix vector multiplication using cluster of Raspberry Pi
- Check the performance of this program by varying the number of nodes in a cluster and plot the graph.
- Implement parallel matrix vector multiplication using cluster of BBB
- Implement parallel matrix vector multiplication using 1-D partitioning formulation
- Implement parallel matrix vector multiplication using 2-D partitioning formulation and compare the performance