ASSIGNMENT 2

Sumedh Masulkar(11736)

Aug 27, 2014

1. Parallel Algorithm

Let us suppose, we have p+1 processors. Where 1 processor will act as a master processor and others as slaves, which will compute products of different parts of matrices. The master will divide the original matrix into smaller matrices and send these matrices to slaves. The slave processors will then return the product of the smaller matrices. The master will finally combine the result from all the slaves and then print the final product.

Consider, √p rows and √p columns. Let P_{ij} represent the processor at i^{th} row and j^{th} column. Without loss of generality, we can assume the matrix to be a square matrix of size n. If it is not a square matrix, we can make it a square matrix by adding additional rows and columns of zeroes. The master processor will partition each matrix into sub-matrices of size n/√p * n/√p each. Let matrix C store the final product.

Then, C_{ij} is computed by processor by processor P_{ij}. Each processor receives all the elements of the rows and columns that are required to generate the n/√p * n/√p elements of C_{ij}. So, the processor P_{ij} computes C_{ij}, and returns it to the master processor.

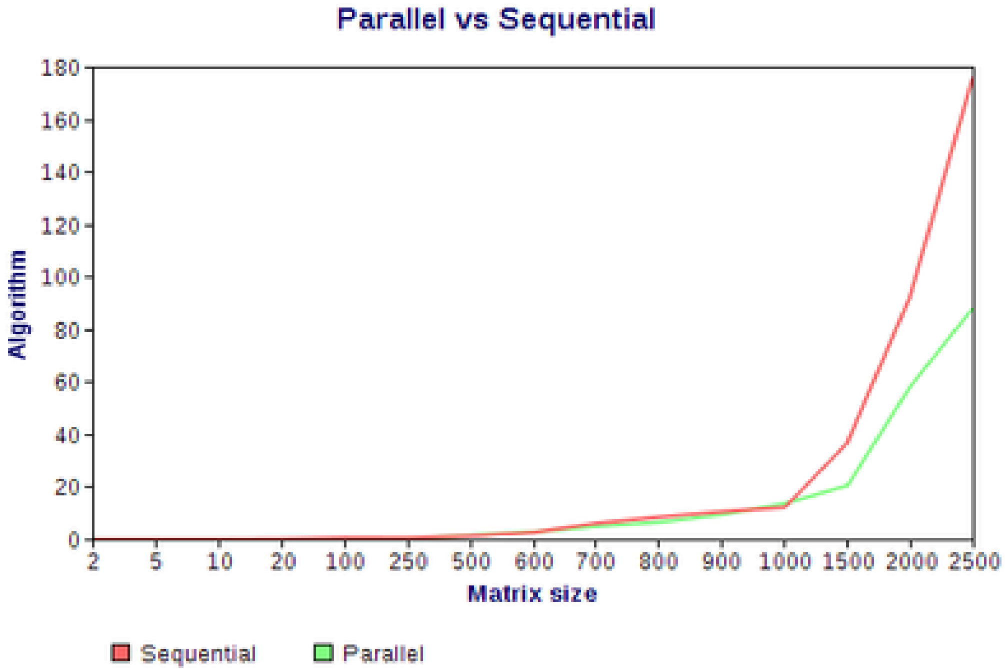
2. Assumption

Given, number of processors is (p+1), then p, which is number of slave processors, must be a perfect square.

3. Generalization

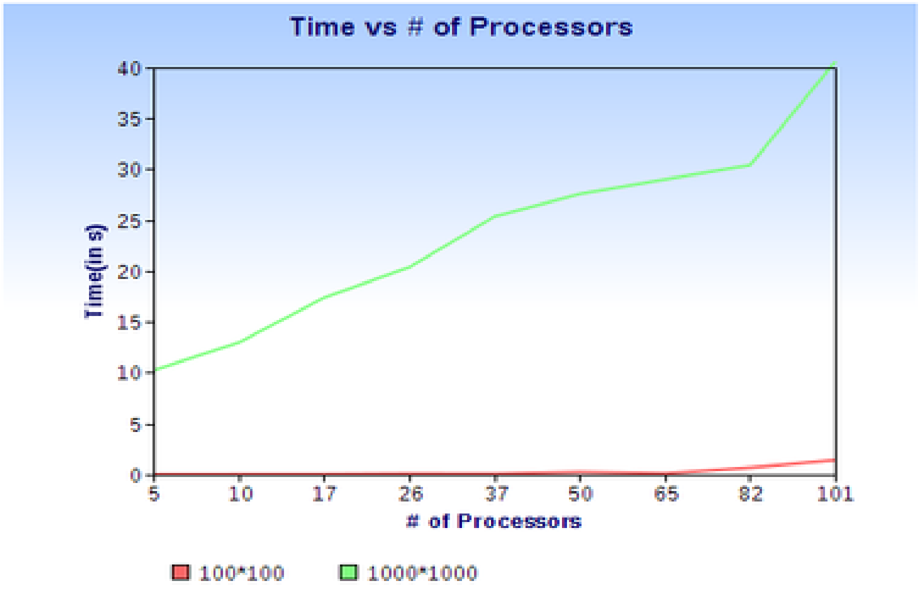
For any 2 matrices A_{m1*n1} and B_{m2*n2}, we will extend it to A_{n*n} and B_{n*n}, where **n** is a multiple of √p and greater than m1, n1, m2, n2. Also, n1 should be equal to m2.

4. Parallel vs Sequential Algorithms



The results have been recorded at processors = 10.
For smaller sized matrices, sequential works faster than parallel, since the overhead of communication increases, but as the size increases, parallel algorithm takes over as workload is distributed evenly over different processors.

5. Variation with number of processors



Conclusion:

• For 100 × 100 matrix size,on increasing processors efficiency always deteriorate as here amount of computation is much less and communication load surpass computation load.

• For 1000 × 1000 matrix size, best results are obtained at number of processor = 5. This is because as you increase processors, workload is distributed but at the same time communication load increases. Therefore, a balance is obtained.