First parallel computing contest: Problem A

Serban Ioan Iorga – serban300@yahoo.com
Computer Science Faculty – "Al I Cuza" University, Iasi, Romania

1. Abstract

This document is intended to describe the method used to solve the problem A featured in the first parallel computing contest. This problem consists in multiplying two matrixes that may contain rectangles of zeros.

2. The problem statement

Given two rectangular matrices, A and B that contain rectangles of zeros (rectangle submatrices with all the elements equal to zero) that may overlap, find the product $C = A * B$. For example the following matrix:

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 0 & 0 & 0 & 2 & 3 \\ 1 & 0 & 0 & 0 & 2 & 3 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 2 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

May have a rectangle of size $3 * 3$ starting in the 2nd row, 2nd column, another one starting in the 4th row, 3rd column, that overlaps with the first one, another one, $3 * 2$, starting in the 4th row, 5th column and another one $1 * 1$ starting in the 6th row, 1st column.

3. Solving the problem

3.1. The data structure and memory access optimization

Given the fact that the matrix contains rectangles of zeros, it may be stored more efficiently than in a classic array. This new storage mechanism would also give us the advantage of having to send less data using MPI.

I have created a new structure which stores each line of a matrix as following: the nonzero elements in an array, and the zero elements as a block, defining only the beginning and the length of the block. For example, the following line:

$$(1 \quad 0 \quad 0 \quad 0 \quad 2 \quad 3)$$

Would be converted to the following structure :

```
Line
{
    Number of nonzero elements = 3
    Numbers = {1,2,3}
    Number of zero blocks = 1
    Zero blocks = {{beginning=1,length=3}}
}
```

While converting the matrices to the new structure, we can also optimize the memory accesses for the multiplications that would follow, by storing matrix B column by column.

3.2. Data sending

Let's consider that n = the size of the matrices and np = number of nodes in the cluster. After transforming the matrices to the new structure, matrix A is broadcasted to all the nodes in the system and matrix B is scattered as follows: each node gets [n/np] columns, and the last node gets n/np+n%np columns. In this way, each node will calculate a submatrix of the resulting matrix C. The resulting submatrices are gathered as classic arrays, because they rarely contain zeros.

3.3. The multiplication

When doing the multiplication, the line from matrix A is reconstructed to a normal array structure and then multiplied with all the columns from matrix B in the corresponding core, stored as described earlier.