MPI implementation of inverse with Gauss Jordan

Gabriel Dimitriu

1 Introduction

Let be A an inversable matrix and his inverse is A^{-1}, then the inverse have the following proprety $A * A^{-1} = I$, where I is the unity matrix.

Through a series of algebraic manipulations, matrix is reduced to a unity matrix and making the same operations for an attached unity matrix. The former unity matrix will be the inverse.

Algorithm 1 *Inverse with Gauss Jordan*

```
procedure inv_gj(mat,inv)
begin
    for k:=0 to n-1 do
        for i:=0 to n-1 do
            if(k<>i) then do
                for j:=k+1 to n-1 do
                    mat[i][j]-=mat[i][k]/mat[k][k]*mat[k][j];
                for j:=0 to n-1 do
                    inv[i][j]-=mat[i][k]/mat[k][k]*inv[k][j];
            else
                for j:=0 to n-1 do
                    inv[i][j]=inv[i][j]/mat[i][i];
                    if j<>i then do
                        mat[i][j]=mat[i][j]/mat[i][i];
                endfor;
                mat[i][i]=1.0;
            endif;
        endfor;
    endfor;
end inv_gj;
```

2 Pipeline communication and Computation

The matrix is scattered to all processor so two consecutive row are in two consecutive processors.

During the k^{th} iteration processor P_k broadcast the k^{th} rows of both matrix to processors all processors. Assume that the processors $P_0 ... P_{p-1}$ are connected in a linear array, and P_{k+1} is the first processor to receive the k^{th} rows from processor P_k. Then the processor P_{k+1} must forward this data to P_{k+2}. However, after forwarding the k^{th} rows to P_{k+2}, processor P_{k+1} need not wait to perform the for with i variable for it's part of matrix until all the processors up to P_{p-1} have received the k^{th} rows. Similarly, P_{k+2} can start its computation as soon as is has forwarded the k^{th} rows to P_{k+3}, and so on. Meanwhile, after completing the computation for the k^{th} iteration, P_{k+1} start the broadcast of the $(k+1)^{th}$ rows by sending it to P_{k+2}. All processors receive p-1 rows before sending their rows to other processors.

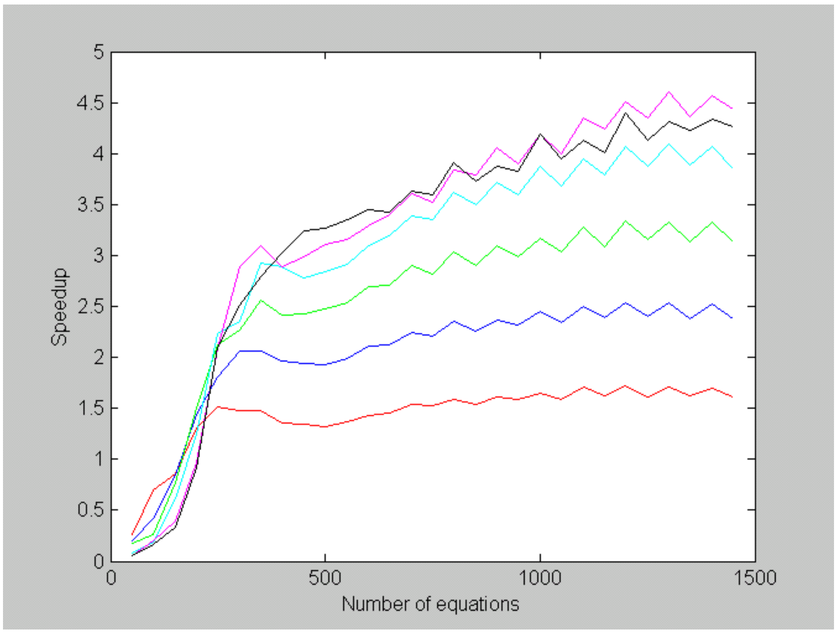
3 Results

The benchmark is made on a 100MBs network with Fedora Core 1 and realteck 8139 network card. The MPI version is MPICH 1.2.6 with native gcc and rsh authorization.

The network consists in 1 PII-500MHz, 2 PIII-800MHz, 1 athlon-1200MHz, 1 PIV-1400MHz and 1 dual PII-500MHz and the speedup is the average of ten runs.

The Inverse with Gauss Jordan is implemented without load balancing so because the fist and in the same time the slowest computer is PII-500MHz so the reference of the speedup is this computer.

In the following graph is presented the speedup with red for two computers, blue for three computers, green for four computers, cyan for five computers, magenta for six computers and black for seven computers.


Speedup
Number of equations