OpenMP Implementation of Gauss Jordan

Gabriel Dimitriu

1 Introduction

Let

$$Ax = b \tag{1}$$

be our system of linear equations, with $A = (a_{ij})_{i,j\in\{1,...,m\}}$, $b = (b_1, ..., b_m)$ and $x = (x_1, ..., x_m)^T$.

Algorithm 1 *Gauss-Jordan*

1. procedure gauss_jordan
2. for k=0 to m do
3. for i=k+1 to m do
4. for j=k+1 to m do
5. $a_{ij} = a_{ij} - a_{ik} * a_{kj}/a_{kk}$
6. endfor
7. $b_i = b_i - a_{ik} * b_k/a_{kk}$
8. endfor
9. for i=0 to k do
10. for j=k+1 to m do
11. $a_{ij} = a_{ij} - a_{ik} * a_{kj}/a_{kk}$
12. endfor
13. $b_i = b_i - a_{ik} * b_k/a_{kk}$
14. endfor
15. endfor
16. for i=0 to m do
17. $x_i = b_i/a_{ii}$
18. endfor
19. end procedure

2 OpenMP implementation

The lines form 3 to 9 are rewritten as

1. for i=0 to m do
2. if(i!=k)
3. for j=k+1 to m do
4. $a_{ij} = a_{ij} - a_{ik} * a_{kj}/a_{kk}$
5. endfor
6. $b_i = b_i - a_{ik} * b_k/a_{kk}$
7. endif
8. endfor

And in front of them I put the following directive which distributed the iterations

#pragma omp parallel for private(j,temp)

We don't need syncronization because the iterrations are idependent.
And the following directive in front of line 16

#pragma omp parallel for

3 Results

I have compile the parallel program with two OpenMP compilers: Omni 1.6 and Intel C Compiler 8.0 for LINUX and the serial with gcc and Intel C Compiler 8.0 for LINUX both with maximum optimization "-O3" and for Intel C Compiler I've put also "-mcpu=pentiumpro -tpp6" for maximum optimization.

The executable were run on a dual pentium II at 500MHz with 256MB RAM and with LINUX Fedora Core 1 and we have the following results wchich are made for a average or 10 runs for serial and parallel programs and with red is plotted the results from ICC and with blue from Omni.