Chirag Jain and Harshvardhan Agrawal

GAUSSIAN ELIMINATION USING OPENMP

Pseudocode:-

```
GAUSS ELIMINATE(U, A, num_elements)
 int i, j, k;
for i := 0 to n − 1 do
            for j := 0 to n − 1 do
U[i*num_elements+j]= U[i*num_elements+j]
 for k := 0 to n − 1 do
            for j := k + 1 to n − 1 do
           A[k, j] := A[k, j]/A[k, k]; /* Division step. */
           end for
           y[k] := b[k]/A[k, k];
           A[k, k] := 1;
           Parallel Region{
           for i := k + 1 to n − 1 do
                      for j := k + 1 to n − 1 do
                      A[i, j] := A[i, j] - A[i, k] × A[k, j]; /* Elimination step. */
                      end for
           b[i] := b[i] − A[i, k] × y[k];
           A[i, k] := 0;
           end for
end for
}
```

Description:-

1.All the elements of the A matrix is first copied to the U matrix

2.Each row is taken one at a time and all the elements of a row are divided by the diagonal element corresponding to the row provided its value is not 0.

3.The parallel region starts here.

4. All the elements of the subsequent rows are then subtracted by the current row value multiplied by a constant which corresponds to the value of the elements below the diagonal element which is now taken to be 1.

5. The parallel region end here.

Note:- It is also possible to parallelize the division of the elements of the rows but it was observed to have a drop in the overall speedup and so was not taken into consideration.

Overall Speedup:-

No of threads	Matrix size	Serial cpu time	new cpu time	speedup
	1024			
2		0.47	0.26	1.807692
4		0.52	0.2	2.6
8		0.54	0.12	4.5
16		0.53	0.1	5.3
	2048			
2		4.61	2.1	2.195238
4		4.6	1.11	4.144144
8		4.47	0.69	6.478261
16		4.99	0.52	9.596154
	4096			
2		43.47	24.93	1.743682
4		50.39	18.26	2.759584
8		41.06	10.99	3.736124
16		41.02	10.25	4.001951
	8192			
2		494.36	235.29	2.101067
4		559.25	144.9	3.859558
8		408.89	91.51	4.468255
16		471.29	103.38	4.558812