Lab3 5451

MingxuanJiang 5588030

April 2020

1

In order to solve Linear Systems of Equations, we first use Gaussian Elimination to transform a linear system into one that is triangular. Then use the parallel back-solve function to solve the triangular system.

To use Gaussian Elimination, I write the pipe_ge function.

1. Initialize some parameters like MPI_Comm_size and MPI_Comm_rank to get the number of processes and myid. Initialize an array temp to store the data of row k.

2. Write a for loop of k to pass the row from 1st to the end.

3. Use if statement to judge whether the row k is in processes myid or not. If row k is in processes myid, using MPI_Send to send row k from array AA to processes South and using MPI_Recv to receive row k from North. If row k isn't in processes myid, using MPI_Recv to receive row k from North. Meanwhile, if processes South needs row k, using MPI_Send to send row k to processes South.

4. Use MPI_Barrier to create a barrier so need to wait all processes finish. It is outside the if statement.

5. In processes myid, compute pivot and calculate elimination through the formula $a_{ij} := a_{ij} - piv * a_{kj}$ for relevant rows (row k+1 to row n+1).

To solve the triangular system, I write the back_solve function.

1. Initialize some parameters like MPI_Comm_size and MPI_Comm_rank to get the number of processes and myid.

2. Write a for loop of k to show the back-solve details.

3. Use if statement to judge whether the row k is in processes myid or not. If row k is in processes myid, get the local index of row k. Solve to get t which means x in the index (x[kloc]).

4. Broadcast t to all processes. Use MPI_Barrier to create a barrier so need to wait all processes finish.

5. Subtract multiple of part of column k. If it is in the processes myid, calculate and update the array A through the formula $A[:,n] -= t * A[:,k]$.

2

Nproc	Timing for GE	Timing for Triangular Solve	Error
1	13.34	6.577×10^{-2}	1.250×10^{-10}
2	6.758	3.732×10^{-2}	1.250×10^{-10}
4	3.603	3.264×10^{-2}	1.250×10^{-10}
8	1.969	2.834×10^{-2}	1.250×10^{-10}
16	1.337	2.797×10^{-2}	1.250×10^{-10}
32	1.707	6.547×10^{-2}	1.250×10^{-10}

We can find that with the increasing number of processes, the smaller timing of both of GE and back-solve will be. If the performance of load balancing is good, the timing of both of GE and back-solve is small. It is no doubt that in the beginning, with the increasing number of processes, the timing will decrease and load balancing will be better. Then the kind of "benefit" is smaller and smaller until achieve overhead. When the number of tasks in each process is small, the weight of overhead increases. So the benefit of parallel is smaller than the impact of overhead. Unnecessary time are added. Last but not least, we can find when using 32 processes, the timing is higher than using 16 processes. It may be because the time of overhead trade off is large.

Nproc	Efficiency of GE	Efficiency of Triangular Solve
1	1	1
2	0.9869	0.8811
4	0.9256	0.5037
8	0.8468	0.2900
16	0.6235	0.1469
32	0.2442	0.0313

The efficiency for both of GE and back solve will decrease with the increasing number of processes. We can find that when nproc = 1,2,4,8 and 16, the efficiency of GE is no less than 50%. And when nproc = 1,2 and 4, the efficiency of back solve is no less than 50%.