Update README.md
Andrew Emerson authored just now

00bca218

README.md 4.19 KiB

HOMEWORK - PARALLEL JACOBI stencil

Resources

All files and instructions can be found from the following gitlab repo:

tccm-barcelona

To download the files on the Galileo100 or Marconi M100, just do the following:

```
git clone https://gitlab.hpc.cineca.it/training/tccm-barcelona.git
```

The files will be in the `Homework/MPI` directory.

Objective: scaling curve of jacobi stencil with MPI

In this exercise we want you run scaling tests of an MPI version of the Jacobi stencil (sometimes called the Laplace algorithm) We provide MPI versions of the program - all you need to do is compile them and run for different sizes of the Jacobi grid, comparing with the serial version (also provided). You can use either the FORTRAN or C versions of the programs on either Galileo100 or Leonardo.

Procedure

1. Compile the serial code

```
gfortran -o serial jacobi-serial.f90
```

or

```
gcc -o serial jacobi-serial.c -lm
```

2. Open an interactive session and run the serial version. For example, on Cineca G100:

```
salloc -N1 -n4 -c4 -A <account> -p g100_usr_prod -t 1:00:00
time srun -n1 ./serial-program 100 100
```

Alternatively use SBATCH commands in a job file (example provided).

The **<account>** has been provided by the demonstrator of the course. Remember, that even if running the serial version you need to use `srun` otherwise the code will run on the login node.

The two inputs of the jacobi code give the grid size, e.g. 100x100. Bigger grid sizes will take more time and take longer to converge.

3. Now repeat with the one of the MPI code versions:
   - using blocking SEND and RECEIVE
   - non-blocking SEND and RECEIVE
   - MPI SendRecv

First compile the program, e.g. blocking MPI

```
module load autoload openmpi
# Fortran
mpif90 -o jacobi-mpi jacobi-block-mpi.f90
# C
mpicc -o jacobi-mpi jacobi-block-mpi.c -lm
```

```
srun -n4 ./jacobi-mpi 100 100       # G100
```

The MPI versions will report the time used automatically.

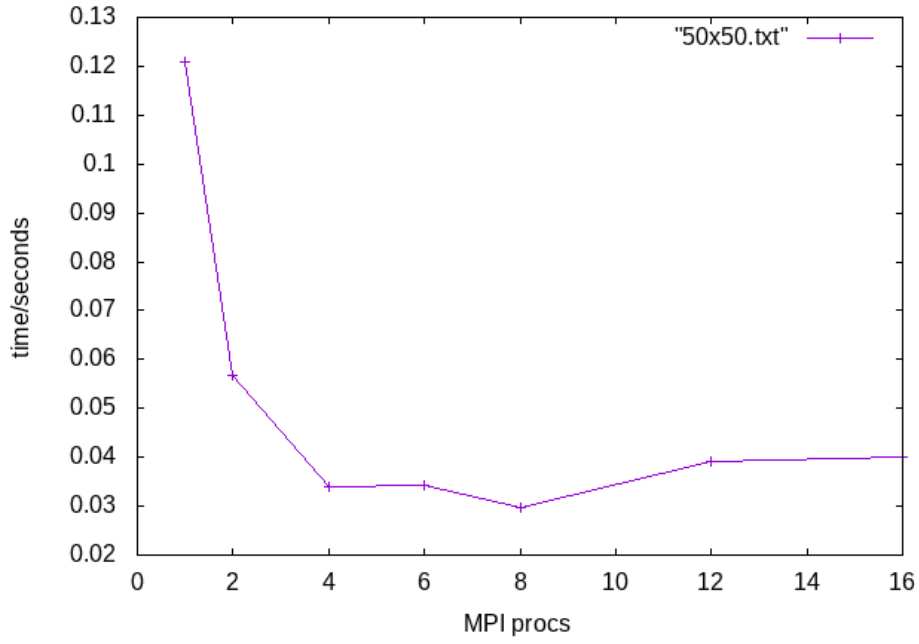
Remember to run the program only via the SLURM batch system!!

4. Scaling curves

You should time the Jacobi MPI for 100x100, 200x200, and 300x300 grid sizes as a function of number of MPI ranks (e.g. 2,4,8,16,.), comparing with the time obtained for the serial version. For each grid size, try :

- Blocking MPI
- Non-blocking MPI
- MPI SENDRECV

An example scaling curve for a 50x50 grid for blocking MPI is given below, showing a maximum scaling at 8 MPI ranks.



Questions

1. What is the *scaling limit*, i.e. the number of MPI ranks corresponding to the maximum performance, for each grid size and for each MPI point-to-point call?
2. How does this change as the grid gets bigger and why?
3. Which MPI version gives the best performance: blocking MPI SEND and RECV, non-blocking MPI SEND and RECV or MPI SENDRECV ? (Do not attempt to explain why - advanced techniques are needed to find out the reason)

Final Comments

- You should check that the parallel calculations give *similar* results to the serial version. In the jacobi example here, we normally obtain numerically identical results (esp. in double precision) but this is not guaranteed in general for parallel programs.

- The 1D-domain decomposition scheme used here, based on dividing the grid into groups of columns or rows, is a simplified example of a 2-D decomposition we would normally used for a jacobi-like stencil. The 2D version would scale much better than the 1D code, particularly for large grid sizes, but the code is more complex and best described elsewhere.

- It is very easy to parallelise also with OpenMP since there are only a few long loops. This code would be a good choice for hybrid MPI+OpenMP: use MPI and domani decomposition to distribute data over shared memory nodes, but use OpenMP to parallise with threads the loops over local data. However, the code needed to implement the MPI+OpenMP hybrid approach is beyond the scope of this course.

Support

For questions regarding you can contact a.emerson@cineca.it or n.shukla@cineca.it. For questions on how to use CINECA resources in general, please contact Cineca user support superc@cineca.it.