Parallel Computer Systems, Fall 2022
Instructions for Databar Exercise 5:
MPI

September 30, 2022

1 Introduction

You will explore MPI and write some smaller distributed MPI programs.

Please read through the entire set of instructions before starting working on the exercises.

You document the work in this exercise in the second mandatory, and individual, report with up to 1.5 pages of text.

2 Learning objectives

During this assignment you will be working towards the following learning objectives:

- Use MPI to write parallel message passing applications.
- You can explain in your own words and in text and with your own commented examples, how message passing can be used to coordinate parallel activities.
- You can write down in text and in your own words definitions or explanations of the following concepts: message passing; message passing model.

3 Reference material

Before carrying out the exercise study chapters 9 and 100 in Hager&Wellein.

4 Setting up the system

You will have to use the HPC machines for these exercises. For instructions on how to connect to the machines using ThinLinc, see `http://gbar.dtu.dk/faq/43-thinlinc`.

Remember to execute `linuxsh` in all console windows before doing any further commands.

A number of files are found on DTU Learn. Use them in the exercises below.

5 DTU's cluster

We will be using DTU's cluster system for the databar exercise (`http://www.cc.dtu.dk/`). The resource manager and scheduler on the cluster is LSF. The command `bsub` is used to submit jobs to the job queue, and the command `bjobs` is used to check the status of jobs. For more information on LSF see `http://www.cc.dtu.dk/?page_id=2534`.

The MPI compiler on the system is `mpicc`. It takes the same arguments as `gcc`, which you previously have been using. You must load the MPI module before the compiler can be used. This is done using the command: `module load mpi`.

6 Submitting MPI jobs

Read through the job script `run.sh`, and answer these questions: Which program will be run? Where will the output be stored? What does the different arguments to the resource manager do? Alter the script to use your study e-mail.

Try to compile a MPI program and submit the job script to the job queue. Hint: use `mpicc` to compile the program, it takes the same arguments as GCC, for example `-o` is used to set the name of the generated executable. Remember to load the MPI module! Use `bsub < run.sh` to submit the program to the job queue.

Check the job status using the `bjobs` command, and wait for it to complete. This might take a minute or two.

Inspect the output in `mpi.log`. Did it match what you expected? What does the output mean? Go back and look at the code, and try to explain the output.

7 Distributed calculations

We are now going back to our calc program from the last databar and we are to write a parallel MPI version of it.

Run your resulting code on different number of processors and produce a plot of speedup versus number of processors up to 16 processors. How do you get statistically sound measurements? For each point what parallel efficiency do you get? What do you think are the underlying overheads leading to the efficiency.

8 License