Lab Course: Distributed Data Analytics
Exercise Sheet 1

Daniela Thyssens
Information Systems and Machine Learning Lab
University of Hildesheim
Submission deadline: Friday May 6, 23:59PM (on LearnWeb, course code: 3116)

Instructions

Please following these instructions for solving and submitting the exercise sheet.

1. You should submit a zip or a tar file containing two things a) python scripts and b) a pdf document.
2. In the pdf document you will explain your approach (i.e. how you solved a given problem), and present your results in form of graphs and tables.
3. The submission should be made before the deadline, only through learnweb.
4. In each lab you have to show your solution works for $P = \{2, 4, 6, 8 \ldots\}$ and provide the timing results (either it is stated in the question or not)

Topic: Distributed Computing with Message Passing Interface (MPI)

In this exercise sheet, you will solve problems using Message Passing Interface API provided by Python (mpi4py). The lecture slides provides the basic introduction to the APIs. In the Annex section there are few useful resources that will help you further understand MPI concepts and provide help in solving following exercises.

Exercise 0: Explain your system

Before we start with parallelizing our code, its always good to mention the system, which will be used for the experiments. It might happen that running time for your program differs on different hardware depending on different factors. Please create a table listing your hardware and software setup. Things you will mention are processor, number of cores, RAM, OS, programming language version information, optimization flags (if used) etc (in future your network connection). This step is important as it gives a clear picture of the physical system used for the experiments.

Second important thing is to time your code. Python provides time package for this task. It is a good idea to just time a section of the code that is actually running in parallel.

Exercise 1: Basic Parallel Vector Operations with MPI (5Points)

Suppose you are given a vector $\mathbf{v} \in \mathbb{R}^N$. Initialize your vector $\mathbf{v}$ with random numbers (can be either integers or floating points). You will experiment with three different sizes of vector, i.e. $N = \{10^7, 10^{12}, 10^{15}\}$. You have to parallelize vector operations mentioned below using MPI API. For each operation you will have to run experiment with varying number of workers, i.e. if your system has P workers than run experiments with workers $= \{1, 2, \ldots P\}$ for each size of vector given above. You have to time your code for each operation and present it in a table. [Note: You have to define/explain your parallelization strategy i.e. how you assign task to each worker, how you divide your data etc.]. You have to use MPI point-to-point communication i.e. Send and Recv.

a) Add two vectors and store results in a third vector.

b) Find an average of numbers in a vector.

Note: Your code should solve a correct problem. To be sure you are doing everything correct, i.e. choose $n = 16$ to verify if your results are correct. Incorrect solutions will not earn you any points.

Exercise 2: Parallel Matrix Vector multiplication using MPI (7 Points)

In this exercise you have to work with a matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ and two vectors $\mathbf{b} \in \mathbb{R}^{N}$, $\mathbf{c} \in \mathbb{R}^{N}$. Initialize the matrix $\mathbf{A}$ and vector $\mathbf{b}$ with random numbers (can be either integers or floating points). The vector $\mathbf{c}$ will store result of $\mathbf{A} \times \mathbf{b}$.

In case of matrix vector multiplication, you will experiment with three different sizes of matrices i.e. $N = \{10^2, 10^3, 10^4\}$. [note: your matrix will be $N \times N$, which means in case 1 you will have matrix dimension 100x100]. You will have to run experiments with varying number of workers, i.e. if your system has P workers than run experiments with workers $= \{1, 2, \ldots P\}$ for each matrix size given above. You have to time your code and present it in a table.

Implement parallel matrix vector multiplication using MPI point-to-point communication i.e. Send and Recv. Explain your logic in the report i.e. how the matrix and vectors are divided (distributed) among workers, what is shared among them, how is the work distributed, what individual worker will do and what master worker will do.

Note: depending on your system RAM you might not be able to run experiment with matrix size $n = 10^4$.

Exercise 2: Parallel Matrix Operation using MPI (8 Points)

In this exercise you have to work with three matrices ($\mathbf{A} \in \mathbb{R}^{N \times N}$, $\mathbf{B} \in \mathbb{R}^{N \times N}$, $\mathbf{C} \in \mathbb{R}^{N \times N}$) i.e each matrix having size $N \times N$. Initialize your matrices $\mathbf{A}$ and $\mathbf{B}$ with random numbers (can be either integers or floating points). Matrix $\mathbf{C}$ will store result of $\mathbf{A} \times \mathbf{B}$.

In case of matrix multiplication, you will experiment with three different sizes of matrices i.e. $N = \{10^2, 10^3, 10^4\}$. [note: your matrix will be $N \times N$, which means in case 1 you will have matrices with dimension 100x100]. You will have to run experiments with varying number of workers, i.e. if your system has P workers than run experiments with workers $= \{1, 2, \ldots P\}$ for each matrix size given above. You have to time your code and present it in a table.

Implement parallel matrix matrix multiplication using MPI collective communication. Explain your logic in the report i.e. how the matrices are divided (distributed) among workers, what is shared among them, how is the work distributed, what individual worker will do and what master worker will do. Perform experiments with varying matrix sizes and varying number of workers. You can look at the implementation provided in the lecture (slide 48) `https://www.ismll.uni-hildesheim.de/lehre/bd-17s/script/bd-01-parallel-computing.pdf`

Note: depending on your system RAM you might not be able to run experiment with matrix size $n = 10^4$.

Annex

1. Palach, Jan. Parallel Programming with Python. Packt Publishing Ltd, 2014.
2. To time your program you have to use MPI.Wtime(): `http://nullege.com/codes/search/mpi4py.MPI.Wtime`
3. MPI4PY tutorial (Python): `https://mpi4py.readthedocs.io/en/stable/tutorial.html`
4. Matrix Operation: `http://stattrek.com/matrix-algebra/matrix-multiplication.aspx`