Department of Computer Science
COMP528 Individual coursework
Assignment 1: MPI Implementation of Pearson Correlation Coefficent

Michael Bane, m.k.bane@liverpool.ac.uk

Overall marking scheme

The coursework for COMP528 consists of four assignments, contributing to 40% of the final mark. The contribution of the single assignments is as follows:

Assignment 1	10%
Assignment 2	10%
Assignment 3	10%
Assignment 4	10%
TOTAL	40%

Failure in any assignment may be compensated for by higher marks in other components of the module.

This document describes Assignment 1, which will be marked according to the following broad criteria:

- correctness and readability of the programs;
- a mandatory report covering design decisions and observed behaviour of the solutions;
- clarity of the arguments explaining the observed behaviours;
- speed of the submitted parallel program and its parallel efficiency and general performance

See detailed marking scheme in the Marking Scheme section below.

Aims of the Assignment 1

- to illustrate the practical aspects of parallel programming using MPI
- to illustrate practical use of an academic HPC system
- to test the skills in analysis of achievable acceleration of programs by parallelization using MPI

Background

This exercise is on programming of some statistical functions. No preliminary knowledge of statistics is required. All necessary concepts are explained in this section. There are no marks regarding knowledge of statistics but marks will be deducted if the equations in this document are incorrectly implemented.

Standard deviation

Standard deviation is a popular measure of variability or diversity of data used in statistics. If we have some set of data, each data point being a real or rational number (e.g. hight of a person expressed in cm), then low standard deviation indicates that the data points tend to be very close to the *mean* value, while high standard deviation indicates that the data points are spread over a large range of values, far away from the mean value.
The standard deviation[1] is defined as follows.
For a set of data points $X = \{x_1, x_2, \ldots, x_n\}$ its mean value is $mean(X) = \frac{x_1+x_2+\ldots+x_n}{n}$. Then the standard deviation of X is the value

$$\sigma_X = \sqrt{\frac{(x_1 - mean(X))^2 + (x_2 - mean(X))^2 + \ldots + (x_n - mean(X))^2}{n}}$$

.

[1] we use here so called population standard deviation, there are other versions, which are not of our concern

Pearson correlation coefficient

Assume now that we have two data sets of *equal* size $X = \{x_1, \ldots, x_n\}$ and $X = \{y_1, \ldots y_n\}$ and we would like to find out if there is some *correlation* between the values x_i and y_i. For example, if x_i and y_i represent the height and the weight of a person, respectively, then, perhaps, there is some correlation. On the other hand, if x_i and y_i represent the height of a person and his/her performance in COMP528, then I doubt that there is a correlation.
There are different ways to measure the correlation between two data sets. Perhaps, the most popular is a *Pearson correlation coefficient*, which is defined as follows.
For two data sets $X = \{x_1, \ldots, x_n\}$ and $Y = \{y_1, \ldots, y_n\}$, their Pearson coefficient

$$\rho_{X,Y} = \frac{\frac{1}{n}\Sigma_{i=1}^{i=n}(x_i - mean(X)) \cdot (y_i - mean(Y))}{\sigma_X \cdot \sigma_Y}$$

where σ_X and σ_Y are standard deviations of X and Y, respectively, as defined above. The Pearson correlation coefficient always has the value in between -1 and 1. The closer it to 1 the more indication of the correlation between data values[2].
If you would like to learn more on the standard deviation and Pearson correlation, the Wikipedia entries are good starting points.

Computing the Pearson correlation coefficient

This exercise asks you to write a program computing Pearson correlation coefficient using MPI.
The input data are given in two arrays a and b of values of type *double*. The length of each array is at least 2,000,000 in order for reasonable timings on today's modern processors. You may also wish to illustrate timings on larger datasets.
You need to write two C programs and run them on the Chadwick HPC cluster in batch. The two required C codes are:

- a serial implementation without parallelism that implements the algorithm for computing Pearson correlation coefficient of data presented in arrays a and b;

[2]notice that Pearson correlation coefficient captures only one type of the correlation that is a *linear dependence*

- a MPI implementation for computing the same, that works on a varying number of MPI processes

For each you should take a given set of input test data (below), compute the Pearson correlation and output both the Pearson coefficient and the wallclock time taken when run on the Chadwick cluster. For the parallel implementation you should also output the number of MPI processes involved in the calculation. By appropriate runs and calculation you should include in your report the parallel speed-up and efficiencies achieved. As well as analysis of the time taken by the whole code, you should consider how well the parallelisaiton of the computation of the Pearson computation scales.
The program should be evaluated using particular initialization of the arrays by the values of the trigonometric function *sine* applied to the indexes, that is $a[i] = sin(i)$ and $b[i] = sin(i + 5)$.
In the case of parallel implementation, this initialization should be done by a single process (say,rank 0).
Given this exercise is to examine parallelism via MPI, in all cases you should turn off other potential compiler optimisations by explicit use of the `-O0` compiler flag

Marking scheme

The submitted solution will be evaluated by the following criteria split in two parts.

Main Part

correctness of the serial algorithm implementation	15%
correctness of the parallel algorithm implementation	35%
programming style	10%
understanding of principles of parallel programming and of MPI	10%
analysis given in the report	10%
TOTAL	80%

and

Bonus part

original contribution, either in the algorithm design, or in the analysis	20%
TOTAL	20%

Submission

You need to submit:

- source code of serial C program
- source code of C program using MPI
- short report with the analysis of your program and its performance, including documentation on how you compiled your code and obtained the timing data

The work must be submitted electronically by going to the Web page at
`https://cgi.csc.liv.ac.uk/login.php`
and follow the link "Coursework submission." This must be done by

16.00 (GMT) on Monday 29 October 2018

Please be aware that the standard University policies

- on plagiarism, collusion and fabricated data
  `www.liv.ac.uk/tqsd/pol_strat_cop/cop_assess/cop_assess.doc`,
  Section 8
  and
- on late submission
  `www.liv.ac.uk/tqsd/pol_strat_cop/cop_assess/cop_assess.doc`,
  Section 6

are applied to this assignment.