University of Wolverhampton
School of Mathematics and Computer Science
Herald College, Kathmandu

Assessment Brief

Module	6CS005 High Performance Computing
Module Leader	Mr Jnaneshwar Bohara
Semester	1
Year	2019-20
Assessment	Portfolio
% of module mark	100%
Due Date	23:59 31 December 2019
Hand-in – what?	Portfolio as specified in this document
Hand-in- where?	On Canvas
Pass mark	40%
Method of retrieval	Resit. Specification of work to be done needs to be negotiated with module leader at feedback session
Feedback	Available from module leader on 1 February 2020
Collection of marked work	At feedback session

This module is assessed by portfolio. Your portfolio will consist of:

- A learning journal that uses the specified template.
- Your source code and data files that use the specified structure.
- It is recommended that you use a revision control system, such as subversion, for maintaining your source code. This will enable your code to be backup up and will also provide a source of evidence in your defence if collusion is suspected. It is recommended that you submit a verbose subversion log obtained by executing:
  svn -v log > svn-log.txt

The module is assessed by evidence that you provide of:
- Analysis of supplied programs and experimentation with them. Capturing results and depicting them as graphs.
- Writing programs to solve computationally expensive problems that require High Performance Computing.
- Written explanations of your results and reflections of your learning.

A grade calculator is provided on Canvas for your use. You should use it to estimate your progress and expected level of achievement.

Three different technologies, POSIX Threads, CUDA and MPI will be combined with 3 application domains, password cracking, image processing and linear regression to enable a variety of application methods to be explored. The following sections are first ordered, by technology then by application domain.

1.1 Password Cracking using POSIX Threads

You have been sent a password cracker program by email. If you have not received the email talk to your tutor immediately.

The program contains a password that has been encrypted using the SHA-512 algorithm. The job of the program is to crack the password, i.e. find the plain-text equivalents. It is known that the passwords consist of two uppercase digits followed by a 2 digit integer. An example password is *JB12*.

Portfolio Tasks

a) Run the program 10 times and calculate the mean running time.

b) In your learning journal make an estimate of how long it would take to run on the same computer if the number of initials were increased to 3. Include your working in your answer.

c) Modify the program to crack the three-initials-two-digits password given in the *three_initials* variable. An example password is JSB99.

d) Write a short paragraph to compare the running time of your three_initials program with your earlier estimate. If your estimate was wrong explain why you think that is.

e) Modify the original version of the program to run on 2 threads. It does not need to do anything fancy, just follow the following algorithm.

```
Record the system time a nano second timer
Launch thread_1 that calls kernel_function_1 that can search for passwords starting from A-M
Launch thread_2 that calls kernel_function_2 that can search for passwords starting from N-Z
Wait for thread_1 to finish
Wait for thread_2 to finish
Record the system time using a nano second time and print the elapsed time.
```

f) Compare the results of the mean running time of the original program with the mean running time of the multithread version.

1.2 Image Processing using POSIX Threads

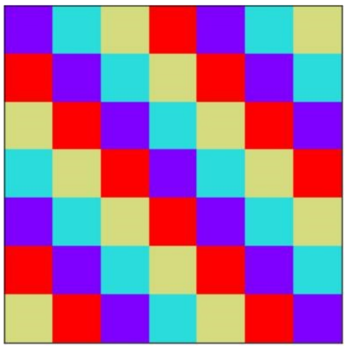
You have been sent an image processing program by email. If you have not received the email talk to your tutor immediately.

The program displays an image alongside another version of it that has been processed using a technique called *edge detection.* The type of edge detection used comes from a suite of algorithms called convolution.
The basic principle is for the algorithm to build the resulting image, pixel by pixel. For each resulting pixel, its value depends on the value of the equivalent in the original image, combined with its 8 adjacent pixels. In this particular usage of convolution several of the components used in multiplications are zero so have been left out of the code to simpilify it. Lecture 3 covers the technique in depth and shows how it can be applied very easily to operations other than edge detection.

Portfolio Tasks

a) Run the program and capture the resulting image to put in your learning journal.

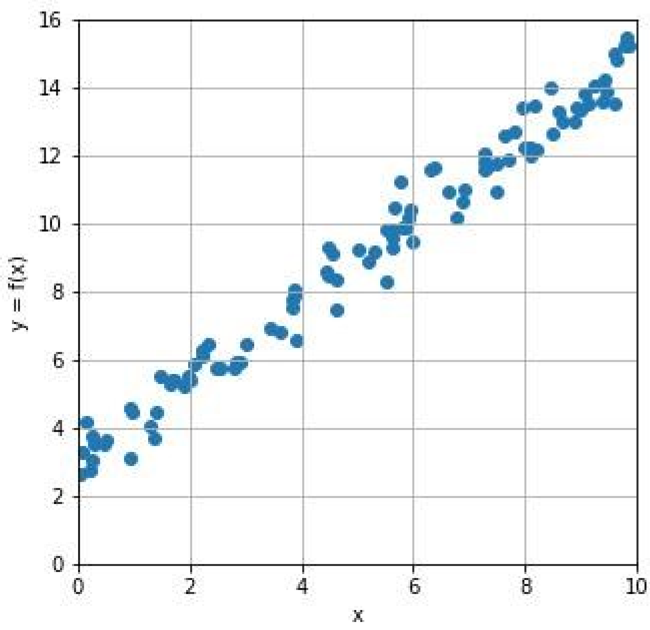
b) Implement a version of the edge detector that can process 4 pixels in parallel. It should do this by using a striding technique, i.e. each thread processes every fourth pixel as shown in the small image below. The first thread processes the blue/purple pixels, the second processes the cyan ones, the third thread processes the yellow ones and the fourth thread processes the red ones.

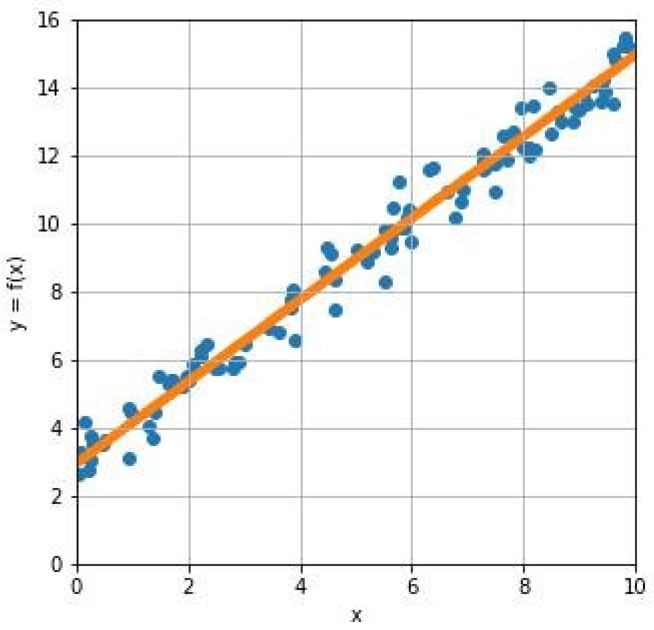


c) Compare the relative running times of the original edge detection program, with the multithread one.

1.3 Linear Regression using POSIX Threads

Linear regression involves finding the equation of the line that best models a set of data points. In the figures below the left graph shows the data points of a linear regression problem. The graph on the right has the solution overlaid.





This is considered to be a minimisation problem where the goal is to minimise the distance between all data points and the line. Minimisation and the gradient descent method of solving it are covered in Lecture 3. The line is defined by the equation *y* = *mx* + *c*, where *m* is the slope and *c* is the intercept with the *y* axis. The goal is to find values for m and c that result in the minimum error.

Portfolio Tasks

a) Do a scatter plot of your dataset and put it in your portfolio.

b) Have three guesses at the optimum values for m and c and plot them.

c) Run the program to see what solution it finds. Overlay the line that was found by the program on to a dataset scatter plot and comment on the solution.

d) Remove any extraneous printf statements from the program and find its mean running time.

e) During each iteration of the program, eight values for m and c are evaluated to measure the resulting error. Create a modified version of the program that performs each of the evaluations on a different thread.

f) Calculate the mean running time of the multithread version of the program and use to make comments on the relative performance of the 2 versions.

2.1 Password Cracking using CUDA

Portfolio Tasks

a) Create another version of the two-initials-four-digits password cracker. It should use separate CUDA thread to process each possible pair of initials, i.e. thread 0 should process AA through to thread 675 processing ZZ. Each thread must therefore perform the 10000 iterations that explore the four digit integers.

b) Compare the mean running time of the CUDA version with the original and multithread versions.

2.2 Image Processing using CUDA

Portfolio Tasks

a) Create another version of the edge detection program that utilises CUDA. Each CUDA thread should process an individual pixel, therefore given an image that is 100 x 72 this will require 7200 threads.

b) Compare the mean running time of the CUDA version with the original program and multithreaded versions. Comment of the effect of invoking more threads than cores.

2.3 Linear Regression using CUDA

Portfolio Tasks

a) Create another version of the linear regression program that utilises CUDA. There should be 1000 CUDA threads that can evaluate a single data point in parallel. The main algorithm therefore needs to call the CUDA kernel 8 times for each minimisation iteration.

b) Compare the mean running time of the CUDA version with the original program and multithreaded versions.

3.1 Password Cracking using MPI

Portfolio Tasks

a) Create another version of the two-initials-four-digits password cracker. A master should share the work out to two compute instances. i.e. instance 1 should explore AA through to MZ, whilst instance 2 should explore NA through to NZ.

b) Compare the mean running time of the MPI version with the original, multithread and CUDA versions.

3.2 Image Processing using MPI

Portfolio Tasks

a Create another version of the edge detection program that uses 4 MPI instances to each process a quarter of the image in a horizontal band. A master instance should share out the work and collate the results back into one image.

b) Compare the mean running time of the MPI version with the original, multithread and CUDA versions.

3.3 Linear Regression using MPI

Portfolio Tasks

a) Create another version of the linear regression program that uses 8 MPI instances, to each compute the error associated with a specific regression line. The master instance should run the main algorithm and use a different instance to compute the error associate with each m, c being explored.

b) Compare the mean running time of the MPI version with the original, multithread and CUDA versions.

Marking and Feedback

Use the grade estimator in conjunction with the marking criteria and feedback sheet to estimate how well you are doing with the module. If you start falling behind tell the module leader.

Submission of work
Your completed work for assignments must be handed in on or before the due date. You must keep a copy or backup of any assessed work that you submit. Failure to do so may result in your having to repeat that piece of work.

Electronic submission
This is normally done via Canvas. Any special instructions will be available on the upload tag or within the assessment brief.

Penalties for late submission of coursework
Standard University arrangements apply. ANY late submission will result in the grade 0 NS being allocated to the coursework.

Procedure for requesting extensions / mitigating circumstances This is done via eVision. Further information can be found at http://www.wolvesunion.org/advice/academic/

Retrieval of Failure
Where a student fails a module (less than 40% for undergraduate modules, less than 50% for postgraduate modules) they have the right to attempt the failed assessment(s) once, at the next resit opportunity (normally July resit period) . If a student fails assessment for a second time they have a right to repeat the module.

NOTE: Students who do not take their resit at the next available RESIT opportunity will be required to repeat the module.

Return of assignments
Assignments will be return during the module's feedback session
If you have any questions regarding your feedback you normally have two working

weeks from the date you receive your returned assessment and/or written feedback

or receive your exam results to contact and discuss the matter with your lecturer.

Cheating
Cheating is any attempt to gain unfair advantage by dishonest means and includes plagiarism and collusion. Cheating is a serious offence. You are advised to check the nature of each assessment. You must work individually unless it is a group assessment.

Cheating is defined as any attempt by a candidate to gain unfair advantage in an assessment by dishonest means, and includes e.g. all breaches of examination room rules, impersonating another candidate, falsifying data, and obtaining an examination paper in advance of its authorised release.

Plagiarism is defined as incorporating a significant amount of un-attributed direct quotation from, or un-attributed substantial paraphrasing of, the work of another.

Collusion occurs when two or more students collaborate to produce a piece of work

to be submitted (in whole or part) for assessment and the work is presented as the work of one student alone. For further details see:
http://www.wolvesunion.org/advice/academic/