|  |  |  |
| --- | --- | --- |
|  | **SCHOOL OF ENGINEERING, TECHNOLOGY & DESIGN**  **ASSIGNMENT GUIDELINES** | |
| **TITLE OF MODULE**:  MCOMD3HPC – High Performance Computing | | **MODULE COMPONENT:**  **50% of Module** |
| **MODULE TEAM**: Dr Vijay Sahota | | **ASSIGNMENT CONTACT**: Dr Vijay Sahota  [Vijay.Sahota@canterbury.ac.uk](mailto:Vijay.Sahota@canterbury.ac.uk) |
| **ASSIGNMENT DEADLINE**:  10th Jan 2022 14:00 | | **EXPECTED FEEDBACK DATE:**  1st Feb 2022  **Location of Feedback:** VIA TURNITIN ON BLACKBOARD |
| **ASSESSMENT TYPE** | | Report/ Digital artefact |
| **Where to Submit:** BLACKBOARD TURNITIN SUBMISSION TOOL  If you experience any problems with this system then please contact the Computing Administration Team ([computing@canterbury.ac.uk](mailto:computing@canterbury.ac.uk))  **What to Submit:**  **A 2000 (Max) word report of high professional standard & Source code+ data.** | | |
| **TITLE OF ASSIGNMENT**: The Threaded Matrix | | |

**ASSIGNMENT INSTRUCTIONS**

This is an individual submission and must be your own work.

There are ***four tasks***. You must complete ***all four tasks***.

**General advice**

* You are required to back up your work regularly onto your N: drive and on removable storage devices. Always check the date-stamp on your files before submission.
* You **must** submit your work using the software versions we currently have on the University’s network.

**Scenario:**

Linear Algebra (LA); has been a staple for many applications of the present. ANN, image manipulation & element modelling are just a few examples.

You have been employed by a HPC solution provider to investigate how to apply their LA toolkit to a HPC environment. To start things off, you have been tasked to investigate and propose solutions to the matrix multiplication problem.

It is envisaged that you implement multiple solutions and then test them under differing conditions; such that, an accurate comparison can be made along with conclusive evaluations on which is the better solutions.

Greater details on solutions and tests are provided in the tasks section.

**Note: It is expected that you produce your work in Eclipse, and as such will submit a zipped version of your project on Black board along with a separate word report in their respective buckets.**

**If any issues arise with submission buckets, email your submission to the assignment contact (Vijay)**

**PLEASE NOTE:**

**All the code you produce must be your own work, and the use of Java built in models/ solutions such as Thread pool/synchronised are not to be used. The use of the Thread class and rand is permitted.**

**Each task should have their own classes/ implementations/ main thread – do not submit one single class that covers all tasks.**

**TASK 1: Simple implementation – gold standard:**

For this section you are to simply implement a matrix multiplication in Java for two 1000 x 1000 matrices of randomly generated numbers. The code must display in the command prompt both the original matrices and their multiplied result.

Note key snippets of code & screen shots should be included in your report.

*(~200 Words)*

[10 Marks]

**TASK 2: Three Models:**

For this section you are to implement three threaded versions of matrix multiplication, where the results will be verified by the gold standard code.

The models you are implement are the following:

* Master Slave
* Loose coupling
* Symmetric

Please note that each implementation must have the ability to set the number of processing elements, but there is always one master/main thread.

For each model implementation, its multiplied output must be verified with the gold standard, a simple subtraction of both results will prove this.

Marks will be awarded for suitable codding models chosen and the implementation of perceived thread safety/ measures taken.

Note key snippets of code & screen shots should be included in your report.

*(~800 Words)*

[40 Marks]

**TASK 3: Testing & Logging:**

For this section you are to test & compare the three models you have created and display the results in a concise way, such that it should be evident which model performs best/ under given conditions. When carrying out your tests you should run the test multiple times and record the average.

Given that you are working for a HPC solution provider, the cost/ speed of execution is the main metric you should be looking to reduce.

However, it will be assumed that the results given are correct – but should be pointed out if & when they become invalid.

When testing, varying the input matrix in size from 100 x100 to 10,000 x 10,000 should be investigated with a fixed processing element count of 100.

The following test will be varying the process element count from 1 – 100.

It is expected you execute pre-tests so that you can decide on ‘good’ testing intervals when carrying both the tests above (making note of findings).

Note this section you are only to present your results, ideally using graphs/ tables.

Note any key snippets of code & screen shots should be included in your report where required.

*(~400 Words)*

[20 Marks]

**TASK 4: Evaluation & Conclusion:**

For this section you are to provide an evaluation of each model, explain/ justifying why they performed the way they did.

Following this you are to recommend a model for use by the HPC solution provider to use in a HPC environment (Clustered compute) – here you will also be expected to justify your selection and its suitability for this environment.

*(~400 Words)*

[20 Marks]

**Quality** of report presentation, assumptions, proper referencing, etc.

[10 Marks]

**Mark Distribution:**

|  |  |  |
| --- | --- | --- |
| **Task** | **Detail** | **Marks** |
| **1** |  | **10** |
|  | Complete implementation & documented in the report. | **5-10** |
|  | Incomplete implementation & minimal documentation in the report. | **0-5** |
| **2** |  | **30** |
|  | Complete and correct implementations, there may be minor errors but the solution provided factors in thread safety/ concurrency. | **10-40** |
|  | Complete implementations but with minor errors, which are documented in the report. | **15-30** |
|  | Minimal – not all three implementations/ or with errors. | **0-15** |
| **3** |  | **20** |
|  | Complete, sufficient data points along with any assumptions made. | **10-20** |
|  | Fair attempt tests carried out but there are faults in accuracy or insufficient data points | **5-10** |
|  | Minimal – none implementation | **0-5** |
| **4** |  | **20** |
|  | Complete attempt made, good justification of models, along with selection – factoring HPC environment & in-depth discussion. | **10-20** |
|  | Fair attempt made good justification of each model’s behaviour along with some attempt (unjustified) selection. | **5-10** |
|  | None - Attempt is made, but poor justification of each model’s behaviour. | **0-5** |
|  |  |  |
|  | **Quality** of report presentation, assumptions, proper referencing, etc. | **0-10** |
|  | **MAX TOTAL** | **100** |

|  |  |
| --- | --- |
| General Guide | |
| **Percentage** | **Description** |
| 75-100 | Complete: Excellent, Complete implementation with documentation to clearly explains all key aspect of function and process along with justification to key aspects (such as how thread safety is reached) |
| 25-75 | Good: Complete implementation with documentation to clearly explains most key aspect of function and process – but some aspects are missing/ un justified. |
| 1-25 | Poor: Incomplete implementation with little documentation. |
| 0 | No Attempt |

**INDICATIVE TIME LINE FOR PLANNING:**

|  |  |
| --- | --- |
| Time Line | |
| **Duration** | **Task/Tips** |
| 1-1.5 days | Time allocated to task 1; Here a simple implementation as setting of your gold standard – you may want to compare your results with other implementation to confirm accuracy. |
| 4-5 days | Time allocated to task 2; you simply need to implement threaded models, set some time aside to research how to tackle this before diving into codding. You may want to use a fast prototyping style – but please keep every version you make, as you need to revert to a prior version. |
| 1-2 days | Time allocated to task 3; here you are simply running multiple runs of your tests & login the results – whilst you are spending time waiting for code to execute you may want to predict behaviour as author of code, this will help with the final task.  You may discover minor errors in your code, if fixable do so or log in your write-up. |
| 1-2 days | Time allocated to task 4; you should ideally not spend more than a day obtaining insight into the HPC environment and what this means for your models, your results and comments on each model & selection should take my the next day. |
| \* | You should be adding to your write up as you go along so as to, at any point have material to submit. The suggested timeframes are indicative, you may want to revisits parts and tweak your work – keep this in mind when you plan for this assessment. |

**FORMATTING OF SUBMISSION:**

You are to upload a zipped version of all your source code onto Bb (Turnitin), a bucket/ link will be provided in the same location where you downloaded this assignment.

You are to upload your written part onto Bb (Turnitin), a bucket/ link will be provided in the same location where you downloaded this assignment.

Any **screen shots, tables, figures, charts, illustrations, etc. will not contribute towards the word count**.

Your work must be adequately referenced throughout, using Harvard referencing style. Pears & Shields (2016) give a complete guide to Harvard referencing. Guidelines on using the Harvard Referencing style are available at:

<https://www.canterbury.ac.uk/library/docs/harvard.pdf>

<https://www.canterbury.ac.uk/students/docs/study-skills/resource-1-Harvard-Referencing-Guide.pdf>

The report must be submitted using the dedicated Blackboard grade centre submission bucket on or before the submission deadline.

**Deliverables:**

You are expected to produce a report of a high professional standard adhering to the guidelines given at:

P:\COURSEWORK\IT\\_Departmental Standards for Students

**LEARNING OUTCOMES ASSESSED (Fully or Partially):**

**Learning Outcomes of this module:**

1. ~~Critically evaluate the role played by HPC in Science, Technology, Engineering and Mathematics~~
2. Critically evaluate some commonly used HPC platforms and parallel programming models
3. Create a high performance solution to a real world problem

**GRADUATE / EMPLOYABILITY SKILLS GAINED:**

This assessment is an opportunity to develop an understanding of the underlying skills to create a HPC application. HPC and understanding the implications in a distributed environment are highly sought after skills, especially since many application/ market edges are gained via big data processing.

All tasks build on skills required in industry, namely researching a problem (analysis), developing/ pitching an idea (conception/ design/ communication) and documentation (though informal).

This assessment has been designed to exercise your abilities to work on your own and progress closer to becoming a proficient self-learner.

**PROGRAMMES OF STUDY:**

BSc (Hons) Computer Science