Cardiff School of Computer Science and Informatics

Coursework Assessment Pro-forma

Module Code: CMT106
Module Title: High Performance Computing
Lecturer: Professor David W. Walker
Assessment Title: Programming with OpenMP
Assessment Number: 1
Date Set: 11 November 2019
Submission Date and Time: 13 December 2019 at 9:30am
Return Date: 31 December 2019

This assignment is worth 15% of the total marks available for this module. If coursework is submitted late (and where there are no extenuating circumstances):

1. If the assessment is submitted no later than 24 hours after the deadline, the mark for the assessment will be capped at the minimum pass mark;
2. If the assessment is submitted more than 24 hours after the deadline, a mark of 0 will be given for the assessment.

Your submission must include the official Coursework Submission Cover sheet, which can be found here:

https://docs.cs.cf.ac.uk/downloads/coursework/Coversheet.pdf

Submission Instructions

All submissions should be via Learning Central unless agreed in advance with the Director of Teaching. Students should submit all their files as a single zip (*.zip) file.

Description		Type	Name
Cover sheet	Compulsory	One PDF (*.pdf) file	[student number].pdf
Report	Compulsory	One PDF (*.pdf) file, including any relevant plots.	report_[student number].pdf
Code	Compulsory	One or more C source files	No restriction
Input image file	Compulsory	One Postscript (*.ps) file	David.ps
Output image file	Compulsory	One Postscript (*.ps) file	DavidBlur.ps
Output data file	Optional	One Excel (*.xlsx or *.xls) file	Spreadsheet of your timing data

Any code submitted will be run on a computer in the Linux Lab (C/2.08) and must be submitted as stipulated in the instructions above.

Any deviation from the submission instructions above (including the number and types of files submitted) will result in a mark of zero for the assessment or question part.

<u>Staff reserve the right to invite students to a meeting to discuss coursework submissions</u>

Assignment

1. Download the sequential code, blur.c, and the input file, David.ps, from Learning Central.
2. Edit blur.c to correctly refer to the input and output files on your file system.
3. Compile and run blur.c several times (say, between 6 and 10 times), noting the time for the execution of the main computational work. Evaluate the average and standard deviation of these times.
4. Parallelize the main computational loops (the *for* loops over row and col) using OpenMP directives, to produce a code named blurOMP.c.
    a. For dynamic scheduling, make timing measurements for differing numbers of threads and chunk size values. The maximum number of threads should be 4 times the number of cores. As in part 3, the times you present should be averages over several runs.
    b. Repeat part (a) for static scheduling.

[8 marks for correct parallel code]

5. Write a report (2 or 3 pages of text, plus figures) that presents and interprets the results of your timing experiments. The report should include:
    a. A description of the hardware and software environment [1 mark].
    b. A description of the timing experiments carried out [1 mark].
    c. Appropriate graphs of your timing experiments [2 marks].
    d. A discussion of the results that demonstrates a qualitative understanding of the timings, and accounts for any unusual features [2 marks].
    e. A short section presenting any overall conclusions, and giving a reflection on what you have learned from this coursework [1 mark].

Learning Outcomes Assessed

1. The ability to recognize the potential for exploiting parallelism to solve a specific problem.
2. The ability to write simple, regular parallel applications using OpenMP

Criteria for assessment

Credit will be awarded as given in the Assignment section above.

Feedback and suggestion for future learning

Feedback on your coursework will address the above criteria. Feedback and marks will be returned by 31 December 2019 via email.

Feedback from this assignment will be useful for you.