

MSc in High Performance Computing

Coursework for Threaded Programming Part 1

The object of this assessment is to experiment with the loop scheduling options in OpenMP. You will be required to conduct some experiments, and submit a report detailing the results of these experiments, as well as the source code you have written.

You are provided with a piece of code which contains two loops which you should parallelise with OpenMP directives. The code measures the execution time for 1000 repetitions of each loop, and includes a verification test for each loop.

The code can be found on the course pages on Learn. You may choose to work with `c/parallel/loops.c` or Fortran 90 `loops.f90` version.

You should use the `-O3` option to ensure a high level of optimisation. You should compile the code, except for the

<https://eduassistpro.github.io/>

Parallelisation

Add OpenMP directives to parallelise the loops in the routine `loop2`. You should parallelise only the outermost loop in each case.

Add WeChat [edu_assist_pro](https://eduassistpro.github.io/)

SCHEDULE clause options

Once you have parallelised the loops, run the code on 4 threads on the back end of `cirrus`, using the following SCHEDULE clause options:

- `STATIC`
- `AUTO`
- `STATIC, n`
- `DYNAMIC, n`
- `GUIDED, n`

where for the latter three cases, n (the chunksize) takes the values 1, 2, 4, 8, 16, 32, 64. From these experiments, determine for each loop the best scheduling option on 4 threads. Using this option (which may be different for the two loops), run the code on 1, 2, 4, 6, 8, 12 and 16 threads.

Submission

You are required to submit the following:

1. A written report.
(Guideline length: 6-8 pages including figures.)
2. Source code.

The deadline for both report and source code is 16:00 on Friday 25th October 2019. Your report should contain:

- a *very short* introduction;
- graphs of the execution time of each loop versus the chunksize for the STATIC, n , DYNAMIC, n and GUIDED, n schedules.
- graphs of the speedup (T_1/T_p) for each loop using the best schedule versus number of threads.
- some text describing *and explaining* the results you obtained;
- some *brief* conclusions.

You should *not* include any background material in your report. Your source code submission should be self-contained, i.e., it should compile and run without any external dependencies. You should use the `make` command to build your program. The `Makefile` should be in the root directory of your submission. The `Makefile` should be able to build the program with the following options: `make`, `make clean`, `make test`, `make submit`, `make help`. The `make submit` command should submit your program to the grading server. The `make help` command should display the following information: `make` builds the program, `make clean` removes the program, `make test` runs the program, `make submit` submits the program, `make help` displays this information.

The maximum
follows:

- Report content and presentation out of 2
- Source code out of 3.