Parallel Programming (English) Lecture Content and Study Guide

作者：Weifeng Liu

组织：Department of Computer Science and Technology, College of Information Science and Engineering, China University of Petroleum-Beijing

时间：Spring semester of 2021-2022, April 19-June 10

版本：2.00

Teaching Assistant：*Wenhao Dai*

目录

第一章 First Week (2021.04.19 15:30 - 17:15) Lecture Content

1.1 Sections One & Two (Introduction of the Course)

The two 45-minute classes will use multiple kinds of materials for introducing high performance computing and its recent development in China and all over the world. How the course will be teaching will be introduced as well. There are three parts:

(1) Part One: Introduce computing, supercomputing and parallel programming; (about 60 minutes)

(2) Part Two: Introduce the self-study content in the next week; (about 15 minutes)

(3) Part Three: Introduce the homework in the next two weeks; (about 15 minutes)

1.1.1 Part One: Introduction to what are computing and supercomputing, and why we need to learn parallel programming, about 60 minutes

Firstly, the requirement of computing and supercomputing technologies is introduced. Several video clips and some performance numbers are given. Also, from the perspective of the history of microprocessor development, three questions are used to discuss why it is necessary to learn parallel programming. This part uses slides to explain.

1.1.2 Part Two: Introduction to self-study content in the next two weeks (Basic knowledge of parallel processors and OpenMP programming model), about 15 minutes.

In the next two weeks, students will need to study some of the following two courses at Bilibili by yourself, with a total length of about 106 minutes and about 336 minutes, respectively.

Bilibili course 1. 新竹清华大学：并行计算与并行编程课程（周志远教授，2018 年秋季学期） https://www.bilibili.com/video/BV1Yt411W7td

（1）Lecture 1C：什么是并行计算（21 分 59 秒） https://www.bilibili.com/video/BV1Yt411W7td?p=3

00:00-08:28, 并行计算的定义

08:38-15:58, 并行计算和分布式计算的区别

16:00-21:58, 为什么需要并行计算

（2）Lecture 1D：为什么我们需要并行计算？（23 分 19 秒）https://www.bilibili.com/video/BV1Yt411W7td?p=4

00:00-12:57, 为什么需要并行计算

12:57-23:18, 并行计算的趋势

（3）Lecture 2A：并行计算机与并行模型的分类（23 分 00 秒）https://www.bilibili.com/video/BV1Yt411W7td?p=5

00:00-14:50, Flynn 的四个经典分类

14:50-23:00, 共享内存和分布式内存

（4）Lecture 3A：“超级计算机” 的概念应如何界定？（16 分 22 秒）https://www.bilibili.com/video/BV1Yt411W7td?p=7

00:00-09:11, 超级计算机的定义以及衡量性能的方式

09:11-11:50, HPL Benchmark

11:50-16:00, 为什么要使用 HPL benchmark?

（5）Lecture 3B：超级计算机的效能（21 分 03 秒）https://www.bilibili.com/video/BV1Yt411W7td?p=8

00:00-06:25, 超算为什么可以算这么快?

06:25-13:25, TOP500 排名

13:25-21:02, TOP500 发展趋势

（6）Lecture 8C：Shared-Memory Programming: OpenMP（23 分 58 秒）https://www.bilibili.com/video/BV1Yt411W7td?p=23

00:00-04:39, OpenMP 编程模型

04:39-08:15, OpenMP 编程例子

08:15-12:26, OpenMP 指导语句

12:26-23:57, OpenMP 线程设置

Bilibili course 2. UC Berkeley CS267 Applications of Parallel Computers (Kathy Yelick et al., Spring 2018)https://www.bilibili.com/video/BV1qV411q7RS

（1）Lecture 1 Introduction（1 小时 24 分 28 秒）https://www.bilibili.com/video/BV1qV411q7RS?p=1

09:56-17:00, Overview

17:52-37:05, Some of the World’s Fastest Computer

    17:52-30:45, Top500 list

    30:35-37:05, figures of computer development

37:05-55:25, Science using High Performance Computing (science examples)

55:25-72:04, Why the Fastest (all since 2005) Computers are Parallel Computers

    55:25-56:50, Limitations

    56:50-72:04, Moore’s Low

72:04-81:25, Measuring Performance

    72:04-73:32, Runtime

    73:32-77:20, Speedup (Amdahl’s Law)

    77:20-81:25, parallel efficiency

81;25-84:27, What’s in this course

（2）Lecture 2 Single Processor Machines Memory Hierarchy and Processor Features（1 小时 21 分 34 秒）https://www.bilibili.com/video/BV1qV411q7RS?p=2

07:00-17:36, Costs in modern processors (Idealized models and actual costs)

17:36-50:45, Memory hierarchies

17:36-27:20, Temporal and spatial locality

27:20-40:14, Basics of caches

40:14-50:18, Use of microbenchmarks to characterized performance

50:18-63:10, Parallelism within single processors

51:18-53:59, Pipelining

55:49-63:10, SIMD units

63:10-80:10, Case study: Matrix Multiplication

63:21-67:40, Use of performance models to understand performance

68:56-71:50, Simple cache model

71:50-74:50, Matrix-vector multiplication

74:19-80:10, Naïve vs optimized Matrix-Matrix Multiply

（3）Lecture 3 Optimizing Matrix Multiply (cont.), Introduction to Data Parallelism（1 小时 23 分 25 秒）https://www.bilibili.com/video/BV1qV411q7RS?p=3

01:28-46:40, Matrix Multiplication

03:30-09:56, L2 recap: memory hierarchies and tiling

09:56-30:28, Cache Oblivious algorithms

30:28-41:05, Practical guide to optimizations

41:06-41:47, Beyond $O(n^3)$ matrix multiply

41:47-45:21, Basic Linear Algebra Subprogram (BLAS)

48:46-82:40, Introduction to parallel machines and programming

49:46-54:30, Parallel Machines and Programming

54:30-82:20, Data parallel

（4）Lecture 4 Shared Memory Parallelism（1 小时 26 分 47 秒）https://www.bilibili.com/video/BV1qV411q7RS?p=4

10:38-20:05, Shared memory parallelism with threads

20:05-25:42, What and why OpenMP

25:42-30:07, Parallel programming with OpenMP

30:07-85:30, Introduction to OpenMP

30:07-32:44, Creating parallelism

32:44-45:14, Caching

45:14-49:39, Synchronizing

51:52-66:20, Parallel Loops

66:20-73:24, Data sharing

73:24-85:30, Tasks

1.1.3 Part Three: Introduction to this week's homework, namely homework 1 (dense general matrix-matrix multiplication, and a performance comparison with OpenBLAS) and homework 2 (quick sort and merge sort, and a performance comparison with qsort()), about 15 minutes

Configure the Programming Environment

It is recommended to install a Linux system, which usually comes with gcc. If one wants to use Microsoft Windows, MinGW (`https://sourceforge.net/projects/mingw`) is generally needed to manually install. After the installation is successful, gcc and OpenMP should be ready for programming homework of this course.

Homework 1. Todo

(1) Run the given serial code of general matrix-matrix multiplication (GEMM). Try to exchange the order of the loops, try to add OpenMP directive `#pragma omp parallel for` to different loops to parallelize them, verify the correctness of the results, test performance, and write them in the report;

(2) Install OpenBLAS. Test the given code that calls GEMM in OpenBLAS, compare the performance with your own parallel code, and write it in the report.

Homework 1. Report Requirements

(1) Test requirements: multiply two square matrices (a matrix with the same number of rows and columns) A and B to get $C = AB$, and the number of rows and columns are both 100-2000 (take a point every 100 intervals), a total of 20 sets of test data, record the running time of each set;

(2) The test method includes at least: (a) the original serial code, (b) several kinds of serial codes with the order of the loops exchanged, (c) the parallel code of adding OpenMP instruction statements on different for loops, (d) the parallel code of OpenBLAS, (e) Original GEMM code optimized in any way;

(3) Submission deadline: April 29, 2022. Submit the report (student ID number + name.pdf) to your teaching assistant.

Homework 2. Todo

(1) Run the given quicksort and mergesort serial codes, find the opportunity to use OpenMP task parallel, add OpenMP instruction statement `#pragma omp task` to parallelize it, and verify the result correctness, test performance, write them in the report;

(2) Call the qsort() function. Test the code of the given qsort() function in the C standard library, compare the performance with your own parallel code, and write it in the report.

Homework 2. Report Requirements

(1) Test the OpenMP parallel algorithms of quicksort and mergesort under the following conditions. The sorting result is required to be in the ascending order, the calculation result should be correct, and the lengths of the input data are 10^1, 10^2, 10^3, 10^4, 10^5, 10^6, 10^7, 10^8,

and a total of eight sets of tests Data, record the running time of each set;

(2) The test method includes at least: (a) running time under different threads, (b) memory space consumption under different threads, (c) performance under different input data, using three different input data: (I) random number sequence, (II) ascending sequence, (III) descending sequence, (d) performance of qsort() code, (e) original sorting code optimized in any way;

(3) Submission deadline: April 29, 2022. Submit the report (student ID number + name.pdf) to your teaching assistant.

Unified requirements for English homework reports

(1) Use Latex to write (Note: do not use Microsoft Word), and submit the pdf file. It is recommended to write homework on `https://www.overleaf.com/`;

(2) Use Python Matlibplot to draw your figures;

(3) The hardware environment of the machine (CPU model, number of cores, frequency, memory capacity, etc.) should be described in the report;

(4) The report language must be English.