For A3, you can enter your times and check other students’ times by clicking here. Include one digit after the decimal point in your timing results: due to the resolution of timers, more digits are not meaningful.

1. Lectures, Tutorials, TAs

Lectures: Monday, Wednesday 8:30–9:20 T13/127
          Friday          10:30–11:20 T13/127
Tutorials: Tuesday 9:30–10:20 T13/105
           Friday          14:30–15:20 T13/105
Office hours Wed 12:30–13:20 ITB 123

Instructor
Ned Nedialkov, email nelialk
ITB 123, x 24161

Teaching assistants
Thomas Gwosdz, email gwosdzt
Curtis D’Alves, email curtis.dalves at icloud.com

2. Introduction

This course is devoted to studying parallel architectures, parallel algorithms and programming. After covering basic concepts, such as parallel programming platforms and models, we shall study parallel algorithm design and the Message Passing Interface (MPI). Our focus will be on designing and implementing efficient distributed parallel programs, and we shall also study shared-memory programming.

- Course webpage http://www.cas.mcmaster.ca/~nedialk/COURSES/4f03/.
  This web page is generated automatically using latex2html.
  If some text does not show correctly (e.g. the tables in Section 13), please consult the PDF file.

3. Learning Objectives

Postcondition A learning objective for a course is something the student is expected to know and understand or to be able to do by the end of the course. The learning objectives for this course are given below. Taken together, this set of learning objectives constitute the postcondition of the course.

1. Students should know and understand
2. Students should be able to
   (a) design and implement efficient parallel programs using MPI and OpenMP
   (b) analyze speed up and efficiency results

Precondition The precondition of the course is the set of university-level learning objectives that the student is expected to have achieved before the start of the course.

The precondition includes knowledge of

- computer architectures
- data structures and algorithms
- operating systems
- C programming

Mapping to attributes with their indicators The table below shows how the course objectives of the course map to the graduate attributes and indicators relevant for this course. These graduate attributes and indicators are a subset of the full list provided by the Office of the Associate Dean (Academic) that are required by the CEAB (Canadian Engineering Accreditation Board.) The numbering used for the graduate attributes and indicators matches that given in the document produced by the Office of the Associate Dean.

1. Knowledge base for engineering
   1.4. Competence in Specialized Engineering Knowledge 1a–1f

2. Problem analysis
   2.3 Obtains substantiated conclusions as a result of a problem solution including 2b recognizing the limitations of the solutions.

3. Investigation
   3.1 Recognizes and discusses applicable theory knowledge base 2a–2b
   3.3 Estimates outcomes, uncertainties and determines appropriate data to collect. 2b

4. Design
   4.3 Proposes solutions to open-ended problems. 2b
   3.3 Estimates outcomes, uncertainties and determines appropriate data to collect. 2b

5. Use of engineering tools
   5.2 Demonstrates an ability to use modern/state of the art tools. 2b

4. Text

Peter Pacheco, An Introduction to Parallel Programming
5. Schedule of Topics

This is a tentative schedule, and it may change depending on how we progress with the material.

<table>
<thead>
<tr>
<th>Week</th>
<th>Topic</th>
<th>Reading</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jan 5</td>
<td>Introduction to parallel computing and parallel hardware</td>
<td>Chapter 1, 2.1, 2.2, 2.3.1–2.3.3, 2.4, 2.5</td>
</tr>
<tr>
<td>Jan 12</td>
<td>Introduction to MPI, collective communications</td>
<td>3.1–3.4</td>
</tr>
<tr>
<td>Jan 19</td>
<td>Caching, performance issues</td>
<td>Lecture notes</td>
</tr>
<tr>
<td>Jan 26</td>
<td>Communication cost, scalability. Parallel program design, tasks, critical path</td>
<td>Lecture notes, 2.6</td>
</tr>
<tr>
<td>Feb 2</td>
<td>Nonblocking communications, understanding communications</td>
<td>Lecture notes</td>
</tr>
<tr>
<td>Feb 9</td>
<td>Introduction to parallel program analysis</td>
<td>Lecture notes</td>
</tr>
<tr>
<td>Feb 16</td>
<td>Midterm recess</td>
<td></td>
</tr>
<tr>
<td>Feb 23</td>
<td>GPU programming</td>
<td>Youtube: OpenACC 1 OpenACC 2, lecture notes</td>
</tr>
<tr>
<td>March 2</td>
<td>OpenMP</td>
<td></td>
</tr>
<tr>
<td>March 9</td>
<td>Data decomposition techniques</td>
<td></td>
</tr>
<tr>
<td>March 16</td>
<td>Advanced MPI</td>
<td></td>
</tr>
<tr>
<td>March 23</td>
<td>Parallel algorithms: matrix multiplication, communicators and topologies, parallel N-body</td>
<td></td>
</tr>
<tr>
<td>March 30</td>
<td>Parallel algorithms: the traveling salesman problem, tree search</td>
<td></td>
</tr>
<tr>
<td>April 6</td>
<td>Review</td>
<td></td>
</tr>
</tbody>
</table>

6. Grading scheme

| Assignment 1 solution | 8% | 21 Jan – 30 Jan |
| Assignment 1 code | |
| Assignment 2 code & plots | 12% | 30 Jan – 11 Feb |
| Assignment 3 | 15% | 25 Feb – 11 March |
| Test 1: covers lecture notes 1–13 | 10% | 13 February, in class |
| Some examples | |
| Test 1, 2014 | |
| Test 2, 2014 | |
| Test 2 | 15% | 30 March, in class |
| Project | 40% | 11 March – 6 April |

7. Lecture notes

This list and corresponding content will be updated as we progress with the material.

(1) Introduction, mpi-greetings.c, openmp-hello.c
(2) Introduction to MPI Code
(3) Collective communications I Code
(4) Caching Code
(5) Interconnection networks
(6) Communication cost
(7) Scalability
(8) Collective communications II
(9) Nonblocking communications Code
(10) Understanding communications Code
(11) Parallel program design. Tasks, critical path
(12) Introduction to parallel program analysis
(13) Example: odd-even transposition sort
(14) OpenACC. Part I, code
(15) OpenACC. Part II
(16) Data decomposition techniques
(17) Array distribution schemes
(18) Parallel N body
(19) Topologies Code
(20) Communicators and topologies: Fox’s Algorithm
(21) The Traveling salesman problem. Parallel distributed tree search
(22) Distributed shortest paths
(23) Advanced point-to-point communications Code
(24) OpenMP

8. Resources

MPI resources
- MPI standard
- MPI: The Complete Reference
- MPI tutorial
- MPI exercises

9. Course Policy

Course-related announcements will be at the course web site. You are responsible for checking it regularly.

Assignments
- The assignments will be due at the beginning of the lectures.
- A late assignment will receive 5% penalty per day.
- You are allowed to discuss the problems from the assignments. However, you must submit your own work.
- Assignments that are very similar may lose half of their marks. Identical solutions to the same problem will receive zero marks.

Missed work
- An MSAF form must be submitted for the following items to apply. Without an MSAF form, missed work receives zero.
• The weight of a missed assignment will be moved to the weight of the tests. For example, if Assignment 1 is missed, Test 1 will contribute 14% and Test 2 will contributed 19% to the final grade.
• If a test is missed, its weight will be moved to the weight of the other test.
• If both tests are missed, then 25% will be lost from the final grade.

Remarking
• Requests for remarking of an assignment or a test must be made within one week after the marked assignment/test is returned.
• Requests that are later than a week will not be accommodated.

Changes
• The instructor reserves the right to modify elements of this course and will notify students accordingly (in class and post any changes to the course website).

10. Academic Dishonesty

You are expected to exhibit honesty and use ethical behaviour in all aspects of the learning process. Academic credentials you earn are rooted in principles of honesty and academic integrity.

Academic dishonesty is to knowingly act or fail to act in a way that results or could result in unearned academic credit or advantage. This behaviour can result in serious consequences, e.g. the grade of zero on an assignment, loss of credit with a notation on the transcript (notation reads: Grade of F assigned for academic dishonesty), and/or suspension or expulsion from the university.

It is your responsibility to understand what constitutes academic dishonesty. For information on the various types of academic dishonesty please refer to the Academic Integrity Policy, located at http://www.mcmaster.ca/academicintegrity

The following illustrates only three forms of academic dishonesty:
1. Plagiarism, e.g. the submission of work that is not ones own or for which other credit has been obtained.
2. Improper collaboration in group work.
3. Copying or using unauthorized aids in tests and examinations.

11. Academic Accommodation of Students with Disabilities

Students who require academic accommodation must contact Student Accessibility Services (SAS) to make arrangements with a Program Coordinator. Academic accommodations must be arranged for each term of study. Student Accessibility Services can be contacted by phone 905-525-9140, ext. 2865 or e-mail sas@mcmaster.ca. For further information, consult McMaster Universitys Policy for Academic Accommodation of Students with Disabilities.

12. Faculty Notices

“The Faculty of Engineering is concerned with ensuring an environment that is free of all discrimination. If there is a problem, individuals are reminded that they should contact the Department Chair, the Sexual Harassment Officer or the Human Rights Consultant, as the problem occurs.”
### Table 1. Students should know and understand:

<table>
<thead>
<tr>
<th>Objective</th>
<th>Below</th>
<th>Marginal</th>
<th>Meets</th>
<th>Exceeds</th>
</tr>
</thead>
<tbody>
<tr>
<td>1a effect of caches in high performance computing</td>
<td>shows no understanding</td>
<td>can perform some analysis, but generally incorrect</td>
<td>can analyze performance and obtain correct formula for the number of flops</td>
<td>meets plus suggest ways to increase performance</td>
</tr>
<tr>
<td>1b communications and communication cost in distributed computing</td>
<td>shows no understanding</td>
<td>shows partial understanding</td>
<td>can analyze communication cost</td>
<td>meets plus can show how to reduce it</td>
</tr>
<tr>
<td>1c, Amdahl’s and Gustafson’s laws</td>
<td>no understanding of these laws</td>
<td>shows some understanding, obtains partially correct results</td>
<td>can apply Amdahl’s but not Gustafson’s laws</td>
<td>can apply them both and obtains correct results</td>
</tr>
<tr>
<td>1d, scalability and efficiency</td>
<td>does not show understanding of scalability and efficiency</td>
<td>shows basic understanding of them</td>
<td>can identify reasons why a program is not scalable</td>
<td>meets plus understands various forms of scalability and speedup</td>
</tr>
<tr>
<td>1e, parallel program design, task graphs, critical paths</td>
<td>cannot describe a simple parallel algorithm</td>
<td>produces partially correct algorithm</td>
<td>correct parallel algorithm</td>
<td>can show alternatives and present arguments about performance</td>
</tr>
<tr>
<td>1f how to analyze parallel algorithms</td>
<td>cannot derive parallel time</td>
<td>partially correct derivation, shows some understanding of how parallel algorithms are analyzed</td>
<td>correct derivation</td>
<td>complete, correct derivation; identifies weak scalability</td>
</tr>
</tbody>
</table>
Table 2. Student should be able to:

<table>
<thead>
<tr>
<th>Objective</th>
<th>Below</th>
<th>Marginal</th>
<th>Meets</th>
<th>Exceeds</th>
</tr>
</thead>
<tbody>
<tr>
<td>2a, design and implement efficient parallel programs using MPI and OpenMP</td>
<td>has produced a working program, but the efficiency is &lt; 50%</td>
<td>efficiency is between 50% and 75%</td>
<td>efficiency is between 76% and 92%, non-blocking and collective communications are used</td>
<td>efficiency is &gt; 92%, non-blocking and collective communications are used; good test cases are developed; the programs are well structured and documented</td>
</tr>
<tr>
<td>2b, analyze speed up and efficiency results</td>
<td>cannot produce such results</td>
<td>results are obtained, but not interpreted correctly</td>
<td>can interpret such result correctly</td>
<td>meets plus can explain why good or poor results are obtained</td>
</tr>
</tbody>
</table>