CISC 879: Advanced Parallel Programming

Project 2

Due: May 21, 2013

1. Introduction

The objective of this project is to gain hands-on experience parallelizing applications for multicore architectures. You will perform the following main tasks for this project.

1A. If you choose a sequential application, you should profile the application and identify the “hot spots” of the program, i.e., those parts of the program where most of the time is being spent. You should then parallelize some or all of these “hot spots.”

1B. If you choose a program containing OpenMP pragmas, you should remove all pragmas and OpenMP calls (i.e., make into a sequential program) and then parallelize some or all of the loops that where originally using OpenMP pragmas.

2. Write a project report discussing your experience parallelizing the applications and describe experimental results obtained running your parallel versions of the applications compared to the sequential versions.

The main goal of the project is to obtain experimental results to answer several research questions regarding the performance implications of running parallel applications on multicore architectures. You will take one large “real-world” application and port it to GPUs or another multicore architecture. You should collect various statistics (including running time) and report how these statistics change depending on the different parts of the application you parallelize.

2. Project Deliverables

This first deliverable consists of meeting with the instructor for a checkpoint. For this checkpoint, you will discuss the code modifications you have made to the application you are investigating. You should be far enough along that you have 20 minutes of code modifications to discuss. Your checkpoint should be done no later than May 9th. You should go over initial results and deliver a first draft of your project report. The checkpoint will constitute roughly 15% of your project grade. If you are having problems (including compiling and/or modifying your application) it is important that your group send an email to discuss this, but this does not constitute a checkpoint.

The second deliverable consists of an original and complete project report discussing experiments and results. The length of the report should be at least 4 pages. The report will consist of identifying and describing the time-consuming regions of the application you are investigating, discussing how you parallelized the code, and showing
performance results comparing your parallel version of the code compared to the original version. The project report will constitute the remaining 85% of your grade.

IMPORTANT DATES
- Checkpoint Due : May 9th
- Project Report Due : May 21st

3. Benchmarks

The table below shows information about three benchmark suites from which you can choose one benchmark to parallelize. These are just suggestions and it is possible to choose an application from a different source other than these three benchmarks suites. A couple of the benchmark suites (BioParallel and MineBench) contain programs that have already been parallelized using OpenMP pragmas. If you decide to start with an OpenMP benchmark, you should first convert the program to a sequential application by removing all the pragmas and OpenMP calls. Then, you should parallelize the application using the particular programming model of the architecture you are targeting, e.g., OpenCL, CUDA, or OpenACC. If you use an OpenMP program, it would be interesting to compare your parallelized version (using OpenCL, CUDA, OpenACC) and the sequential version to the original OpenMP version. The simplest way of doing this is to use the GCC or the Intel compiler (ICC) and run the original OpenMP on one of the multi-core machines available to the class.

<table>
<thead>
<tr>
<th>Category</th>
<th>Benchmark Suite</th>
<th>Website</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bioinformatics</td>
<td>Biobench, BioParallel (uses OpenMP)</td>
<td><a href="http://www.ece.umd.edu/biobench/">http://www.ece.umd.edu/biobench/</a></td>
</tr>
<tr>
<td>Data-mining</td>
<td>MineBench (uses OpenMP)</td>
<td><a href="http://cucis.ece.northwestern.edu/projects/DMS/MineBench.html">http://cucis.ece.northwestern.edu/projects/DMS/MineBench.html</a></td>
</tr>
<tr>
<td>Financial</td>
<td>Quantlib</td>
<td><a href="http://quantlib.org/">http://quantlib.org/</a></td>
</tr>
</tbody>
</table>

Note: The rest of this document is similar to the Project 1 description, although there are some important changes that have been made so you should read the rest carefully.

4. Project Report Guidelines

Papers must be well written and formatted correctly. We expect you to clearly describe the project and experimental results. You are required to present experimental results using graphs. Also, you are required to use the ACM proceedings format that is provided in the following link (the same as in Project 1).

http://www.acm.org/sigs/publications/proceedings-templates
5. Steps in Designing and Conducting the Experiments

Before running experiments, you should come up with a set of research questions you want to answer. Developing and selecting a concise set of research questions (see sample research questions below) will help you focus on the experiments you need to run.

1. Justify your selection of program you chose in terms of conducting an unbiased experiment with results others can reproduce and (hopefully) that can generalize to other programs.
2. Develop an experimental procedure. Try the procedure (e.g., parallelization) one part of the program and refine the procedure based on this study.
3. Decide how to present the results. What will be the columns and rows of the data tables? What kinds of graphs would help illustrate the results to easily allow drawing conclusions to the research questions? What will the axes of the graphs be? What form will the data be shown in, e.g., bar graph, scatter plot, line graph, box and whiskers graph, etc.? Look at graphs from other papers as examples.
4. Write a draft of the report based on the above steps, including an experimental methodology and results section that contains a place for tables and graphs where you will eventually put your results. You should conduct peer editing among the members of your team and create different versions of this document. As soon as you have a draft, show it to the instructor so he can provide feedback.
5. Given any revisions from Step 2, conduct experiments on the program you parallelized and collect data.
6. You can use a variety of different tools to construct graphs. I suggest using the same graphing tool(s) you used for Project 1.
7. You should conduct an analysis of the data and make observations about various trends. This analysis and observations will become part of your results discussion. Often while discussing results amongst your team, you will identify additional experiments you should have run. Run any additional set of experiments if necessary and continue iterating this process until you are satisfied.
8. Finish writing the complete experimental study report

Sample Research Questions:

1. What impact does each portion of parallelized code have on the performance of your program?
2. Which parallelized portions of your application have the most impact on performance?
3. If you have tried two different compilers, how does performance of the first compiler compare to the performance of the second compiler?
4. How does the program scale when varying the number of cores for the multi-core or threads and thread blocks for OpenCL or CUDA?
Format of Experimental Study Report:

1. **Title, Authors, and Abstract** – This report should look like a conference paper you are going to submit, including a properly formatted title and author names and affiliations. Give your paper an interesting title and include a brief abstract.

2. **Research Question(s)** – The paper should include a concise statement of the research questions and why these are important to investigate.

3. **Variables and Measures** – You should include a description of the independent and dependent variables and how you measured the dependent variables.

4. **Program** – Include a description of the program name, original lines of code (LOC), the LOCs after you parallelized the program, and a short phrase describing what the program does. Also, include a paragraph on the program you chose and why you chose it. Do NOT include more than a paragraph on what the application does. Also, do NOT describe the architecture in any detail unless it pertains specifically to the experiment results in the paper.

5. **Experimental Methodology** – Your paper should describe how you generated the results. For example, include details such as what compilers and other tools were used (e.g., gprof), how many times you ran your parallelized program, and how measurements were taken (e.g., what was used to time program execution?). Specify enough details so that someone else could replicate your experiment. You can also show short code segments that you decided to parallelize.

6. **Results and Discussion** – This section includes tables, graphs, and an explanation of each table/graph starting with what data it contains and how to read it and any observations you can draw. This discussion pertains to your original research questions you embarked to ask and how this data answers those questions.

7. **Related Work** – You should include a description of related work that is relevant to this study. Summarize briefly (two or three sentences) this work and how it relates to your study.

8. **Conclusions** – Finally, your paper should end with a summary of the experimental results and any conclusions you made from the experiments.