CSCI503: Parallel Programming

Final Project Proposal

Revision: 1.0 – December 27, 2012
Due: March 11, 2013 1:59 PM

1 Proposal for Final Project

The goal of this assignment is to help you define and narrow the scope of your final project. In the final project you will design, implement and evaluate the performance of a parallel algorithm or a parallel formulation of a sequential algorithm. The project should be done by small teams of students, with 3 or 4 students per team.

For this assignment each team of students must submit a single proposal. The proposal should be a document (of up to 2 pages) describing:

- The problem to be addressed (sequential or parallel algorithm) and what motivated you to choose it (up to 1 page);
- The approach for paralllelizing/implementing this algorithm, including the programming model/language and parallel machines you intend to use. Discuss why your choices are adequate or relevant to the problem and what are the implementation challenges (up to 1 page).

We will evaluate the proposals and provide feedback on the scope of your project. As a general guideline, the proposed problem or algorithm should not be trivial, that is, it should have at least one of the characteristics of problems that require parallelization, such as:

- High computational intensity (that is, lots of work);
- Very large data sets (that is, data does not fit in memory of a single node of a state-of-the-art parallel machine).

Note that we do *not* expect you to work with very large data sets in your implementation, or with computations that take days to complete execution, due to memory and cpu-time constraints on the available resource. Instead, your implementation should assume that your data does not fit in the memory of a single node, or that the total computation time of a sequential implementation is unacceptable and needs to be improved.

We encourage you to choose a project related to your current or future research or work. The following list includes example projects from previous courses and some suggestions

as well:

- Hybrid-implementation of Gaussian Elimination using both OpenMP and MPI. For example, use OpenMP to spawn multiple threads running on a multicore node with shared memory, and MPI to communicate among machine nodes.
- New implementation of Gaussian Elimination, Conjugate Gradient or another non-trivial problem on GPUs, using CUDA or OpenCL. Performance comparison of different mappings of computation and data to GPU architectures.
- Data-mining algorithms using MapReduce.
- Comparison of parallel implementations of a non-trivial problem on different programming models/target machines. For example, parallel implementations of a sparse-matrix or graph algorithm using MPI and MapReduce, or OpenMP and cloud computing.

2 Submission

Each team should submit a short (up to 2 pages) proposal via Blackboard. Please include the names and USC ids of all team members in your submission.