15-740 Project Proposal

Ravishankar Krishnaswamy (ravishan@cs.cmu.edu)
Harsha vardhan Simhadri (harshas@cs.cmu.edu)

November 1, 2009

1 Web page

Project website will be located at `http://www.cs.cmu.edu:/~ravishan/15740`.

2 Project description

Thread schedulers play an important role in the cache performance of (nested) parallel programs on multicore architectures. While the case of schedulers for completely shared or completely distributed cache systems is reasonably well understood both in theory and practice, it is not clear how threads are to be scheduled on a more complex system of caches. In this project, we want to explore schedulers for such systems. More particularly, we intend to focus on systems such as (tree-like) hierarchy of caches, and possibly caches shared through grid-like interconnects. While it is not certain that future cache hardwares may look exactly like a tree, it is believed that this topology seems to be a good logical approximation. Hence our focus on scheduling threads for such cache structures.

While the general goal is to find schedulers that (a) yield good run times, (b) use less memory bandwidth and (c) are cheap to implement for a wide variety of programs, we believe cache oblivious algorithms are a suitable starting point because of their cache friendly behavior. Restricting our focus to such well behaved algorithms might also help us make more precise statements about the design of schedulers (100% goal). As for the schedulers, multi-level extensions of Work-stealing or PDF schedulers seem to be a natural starting point, as these schedulers are well understood for the simpler cases mentioned above. If time permits, it might be interesting to explore schedulers that can handle varying cache size availability and what kind of replacement policies suit multicore applications (and if these two problems are related). This would be our 125% goal. If things do not go as planned and we do not get to identify one near perfect solution, we hope to have at least studied some schedulers and their relative merits.

3 Logistics

Due to the short time span of the project, we plan to use simulations to measure performance. We are currently looking for a Pin like tool that, in addition to providing information about data accesses, supports multi-threaded programs and allows us to define the schedule. We are still trying to find the correct tool that fits our purpose.

4 Time line/ Milestones

1. **Week 1:** We plan to lay out several reasonable candidates for scheduling policies and find the best simulation tool.
2. **Week 2:** Write an instrumentation tool to collect memory traces for several commonly used parallel programs.
3. **Week 3:** Set up the traces and code in our schedulers and algorithms.
4. **Nov.** 17 **Milestone:** We plan to have tried out a few ideas and have generated some experimental data.
5. **Final Weeks:** Collecting and summarizing data; Making final observations and conclusions.

The critical path in the initial stages seems to be setting up simulations tools, of which we have limited knowledge.

5 Literature search / Getting Started

We have a fairly good idea of the theoretical and simulation studies of the schedulers mentioned above. We also know suitable cache oblivious algorithms for most algorithmic problems. However, we are not sure about simulation techniques and plan to read more about it. We have just looked at a paper that talks about a Pin based tool called CMP$im, but we are not sure if it is possible to program scheduling logic in to it. We hope that discussions with some of the authors on earlier papers will help us decide the right tool.