Project Milestone Report

Frank Fan (wfan)

Major Changes

I have changed my project from the originally proposed cluster scheduling one to the current one, investigating the effects of process control in the context of modern processor features. The project that the original project would be continuing has not been working out and was going nowhere, I was also losing motivation to work on the project. The updated proposal starts on the next page.

Accomplishments and First Milestone

I have read and understood most of what process control entailed in the 1990s, and what results process control brought about with older CPUs. I have also started investigating the dynamic process control mechanism (and its source code) that is currently enabled by default in OpenMP. This was the intended first milestone in the proposal.

Surprises

There were no surprises.

Revision to Milestones

See attached new proposal.

Resources Needed

I do not need any additional resources at the moment.

Revisiting Process Control with New Processor Features

Weihang Fan (wfan)

Project Webpage: http://www.contrib.andrew.cmu.edu/~wfan/300/

Project Description

My collaborator will be Professor Todd Mowry, professor in the Computer Science Department.

This project would involve first measuring the effects of some modern processor features, such as TurboBoost on various workloads. Specifically, we want to look at the behavior (in terms of performance) of programs when they are given *less* threads than cores available. Then, we would need to characterize the kind of workloads for which intentionally decreasing the number of threads below the number of cores would increase performance. To concretely implement a solution taking advantage of this characterization, we will implement a modification to the process control mechanism in OpenMP to dynamically adjust the number of threads based on measured program behavior (e.g. cache misses).

The main challenge of this project lies in the fact that the hypothesis goes against decades of established knowledge in parallel computing - that in general it is best to run parallel programs with the number of threads equal to the number of virtual cores available. In addition, on the implementation side, OpenMP implementations are very low-level and the source codes are complex, figuring out how to modify them to perform process control could be very difficult and nearly impossible. In addition, measuring things like cache misses and processor frequency could be inaccessible to user-mode programs like OpenMP, and we might need to figure out some creative ways of implementing the dynamic measurement.

Project Goals

75%

- Measure the impact, in terms of performance and energy efficiency, of reducing the number of processes on non-false-sharing programs and programs that do have a lot of false sharing
- Characterize program behavior where reducing the number of threads would improve performance

100%

All of the above, and either

- Implement a modification to OpenMP that dynamically detects the number of cache misses and performs process control to improve performance

or

- Implement a modification to OpenMP that dynamically detects when TurboBoost would be more effective than multiprocessing and performs process control to improve performance

125%

- Both of the above

Milestones

1st: Conduct background reading on the subject of process control, investigate how modern frameworks like OpenMP implement process control
2/1: Conduct initial experiments relating number of threads to performance for various types of workloads on the latedays cluster
2/15: Conducting further experiments to characterize workloads that benefit from process control, implement hello-world modifications to OpenMP's process control mechanism
3/1: Start working on the OpenMP process control modification, implement some sort of method of gauging workload characteristic
3/22: Start implementing process control based on the above mechanism
4/5: Continue the above task
4/19: Finish OpenMP modification, debug so it's ready for measurement

5/3: Conduct measurements on OpenMP modification on the same workloads used in initial measurements

Literature Search

I have read multiple papers in the field of process control, including the following:

- A. Tucker and A. Gupta. 1989. Process control and scheduling issues for multiprogrammed shared-memory multiprocessors. In Proceedings of the twelfth ACM symposium on Operating systems principles (SOSP '89). ACM, New York, NY, USA, 159-166. DOI: https://doi.org/10.1145/74850.74866
- Rohit Chandra, Scott Devine, Ben Verghese, Anoop Gupta, and Mendel Rosenblum. 1994. Scheduling and page migration for multiprocessor compute servers. In Proceedings of the sixth international conference on Architectural support for programming languages and operating systems(ASPLOS VI). ACM, New York, NY, USA, 12-24. DOI=http://dx.doi.org/10.1145/195473.195485
- Ben Verghese, Anoop Gupta, and Mendel Rosenblum. 1998. Performance isolation: sharing and isolation in shared-memory multiprocessors. In Proceedings of the eighth international conference on Architectural support for programming languages and operating systems (ASPLOS VIII). ACM, New York, NY, USA, 181-192. DOI=http://dx.doi.org/10.1145/291069.291044
- Thomas E. Anderson, Brian N. Bershad, Edward D. Lazowska, and Henry M. Levy. 1991. Scheduler activations: effective kernel support for the user-level management of parallelism. In Proceedings of the thirteenth ACM symposium on Operating systems principles (SOSP '91). ACM, New York, NY, USA, 95-109. DOI=http://dx.doi.org/10.1145/121132.121151

Resources Needed

In order to perform performance will use the latedays cluster, which has nodes with two six-core Xeon e5-2620 v3 processors with all of the modern features that we would like to measure (TurboBoost, hyperthreading, cache hierarchy, etc.). If needed, we might want to utilize bare metal servers on vultr.io for more variety in hardware.