Brendan Kerr

Professor Saule

Performance Monitoring: Likwid and Intel

Explicitly defining the actual performance in modern processors is a difficult task. As programs become more complicated, it can become difficult to understand where bottle necks exist and exactly what can be done to mitigate said bottlenecks. The internals of processors have become extremely complex with memory being shared, multiple instructions being executed, and multiple cores being utilized. It’s a wonder that any accurate metric can be collected at all. In starting the project, we set out to gain a few different accurate measurements from a select processor in terms of efficiency, speed, and sheer power. When defined in the domain of the project, these measurements translate to latency, bandwidth, and floating point operations per second (FLOPS). A tool was used to help collect these measurements by the name a LIKWID, an acronym that stands for ‘Like I Know What I’m Doing,’ an accurate depiction of how I felt when attempting to achieve said metrics.

As I stated before, it is hard to get accurate metrics from a processor. However, in order to be more confident in our measurements, a few techniques were employed. These techniques include pinning a thread to a CPU, only monitoring the events of a chip at certain key points in execution, and affirming our output measurements with benchmarks published by the creators of the processors used. In pinning a thread to a CPU, we could be confident that the instructions being executed on that thread were local to that core and not being shared or shuffled around, possibly through some shared component like the L3 cache. Waiting to start monitoring the performance of a piece of software until it is absolutely necessary also gave us more confidence in our measurements in that we were aware of what to expect. If instead the performance monitoring tool was used at the beginning of runtime, undesired instructions would most likely get mixed into our measurements, resulting in less accurate output.

The performance monitoring tool had two specific APIs that were most helpful in obtaining metrics. These APIs were the marker API and the standard likwid API. Each offered their own flavor of fine-tuned performance monitoring, but the marker API proved to be especially interesting as it allowed the user to define multiple “marker tags” that would end up tagging those specific measurements. This would in turn give the user the ability to establish different groups of measurements through a tagging scheme. Additionally, the marker API would allow for multi-threaded performance monitoring through OpenMP, a piece of middleware that abstracts multithreaded environments.

As per any projects, a few bumps were hit in the road while trying to accomplish our goal. Specifically, around the instructions allowed through different flavors of processors and compatibility of likwid with said processors. Because we had a specialized goal in mind, we had to end up using specialized variables and function calls to force the processor to do exactly what we wanted (or close to it). Theses specialized tools provided access to specific intel instructions. However, before using certain ones, it was important to ensure that the processor we were testing on had the ability to execute the instructions used within the programs. To do this, one would simply *cat /proc/cpuinfo,* and integigate the ‘flags’ field, which will end up displaying all the available instruction sets along with other valuable information. This proved to be troublesome because some instructions we were hoping to use weren’t compatible with the processor being tested on.

Although we did hit some blockages, I feel as though great progress was made in terms of me being able to further understand how processors work and most importantly their strengths and limitations. One often takes for granted how much a computer is actually accomplishing for its user at any given moment and being able to almost talk to the processor on a more intimate level was very fun. I feel as though I would like to continue in dabbling with performance monitoring in my spare time with a few processors I have along with monitoring a new build I plan to start during the new year.