In-situ performance profiling by utilizing the "unused core"

Tianya Wu
tianya@cspp.cc.u-tokyo.ac.jp
The University of Tokyo
Bunkyo, Tokyo, Japan

Jun Kudo
The University of Tokyo
Bunkyo, Tokyo, Japan

Toshihiro Hanawa*
hanawa@cc.u-tokyo.ac.jp
The University of Tokyo
Bunkyo, Tokyo, Japan

1 INTRODUCTION

In recent years, the number of physical cores built into CPUs has been increasing. However, for HPC cluster, introducing as many cores as possible for computation does not necessarily lead to maximum performance. Thus, there are cases when only a fraction of cores are used for computation can provide optimized overall performance. The cores not involved in the main computation are usually called "unused cores". In this study, we proposed a framework that can utilize the unused cores for profiling during execution with low overhead.

2 UNUSED CORE

Although increasing the number of cores can improve the performance of multi-processor, the ideal performance may not be achieved when all cores are simultaneously activated for computation. Lack of parallelism, memory-bound constraints, as well as power and energy constraints are the reasons for this phenomenon. To achieve maximum overall performance, a fraction of cores are expected to stay idle during the computation. These "unused cores" can be further utilized in the framework. For example, providing some functions like monitoring during execution, auto-tuning, cache prefetching can be very helpful.

3 PROFILING OF THE MAIN COMPUTATION

3.1 SystemTap

In order to realize the monitoring during main computation and the insertion of processing, our framework will use SystemTap as function hooking on the unused cores. In this study, the profiling and processing are introduced when SystemTap detect entries and exits of specific functions as shown in Figure 1.

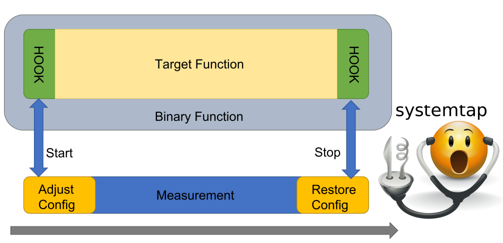

Figure 1: The framework of SystemTap

3.2 PAPI

PAPI is a portable interface to hardware performance counters on modern microprocessors. For profiling, we can use PAPI to collect performance metrics like instruction counts, clock cycles, cache misses, power consumption, which can be important references for some optimizing processing.

3.3 Implementation

For evaluation, we run the experiments on our testing clusters. Table 1 shows the specifications of the system.

Table 1: Experiment Environment.

CPU	Intel(R) Xeon(R) Platinum 8260L
Number of cores	24 x 2
Frequency	2.4 GHz
Compiler	Intel C++ Compiler v19.0.5.281

Figure 2 shows that using 42 out of 48 cores for computation can provide the optimized overall performance. If we utilize the remaining "unused cores" for profiling by SystemTap and PAPI, the increased overhead is about 0.15% as shown in Figure 3, which is relatively low and shows potential for the future extension.

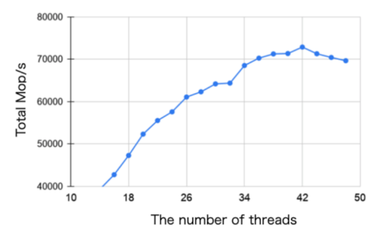

Figure 2: Total Mop/s over different number of threads

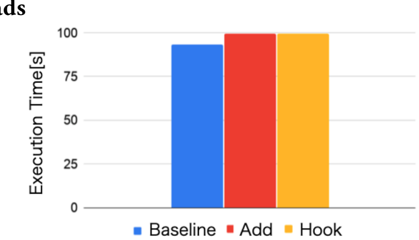

Figure 3: Average execution time over different execution pattern

4 CONCLUSION AND FUTURE WORK

In this study, we realized in-progress profiling of the main computation with low overhead by utilizing the "unused cores". For future work, we would like to add some optimization processing in our framework such as dynamically adjusting the parallelism as well as thread affinity of the main computation based on the result of profiling.