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

Tianya Wu Jun Kudo Toshihiro Hanawa

The University of Tokyo, Kashiwa, Chiba, Japan

Acknowledgment: This work was partially supported by JSPS KAKENHI Grant Number JP20H00580.

Introduction

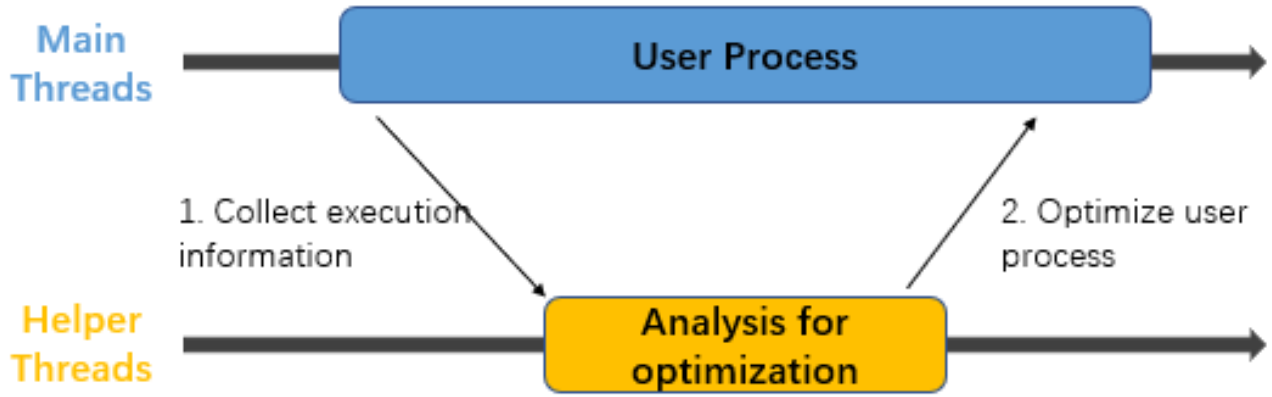
- In recent years, there are more and more “unused” cores in HPC clusters.

- We considered a framework called UTHelper to leverage the “unused cores” to provide some supportive function to the main computation.

- Dynamic profiling during the execution of main computation can be achieved on the “unused cores”.

Proposed framework

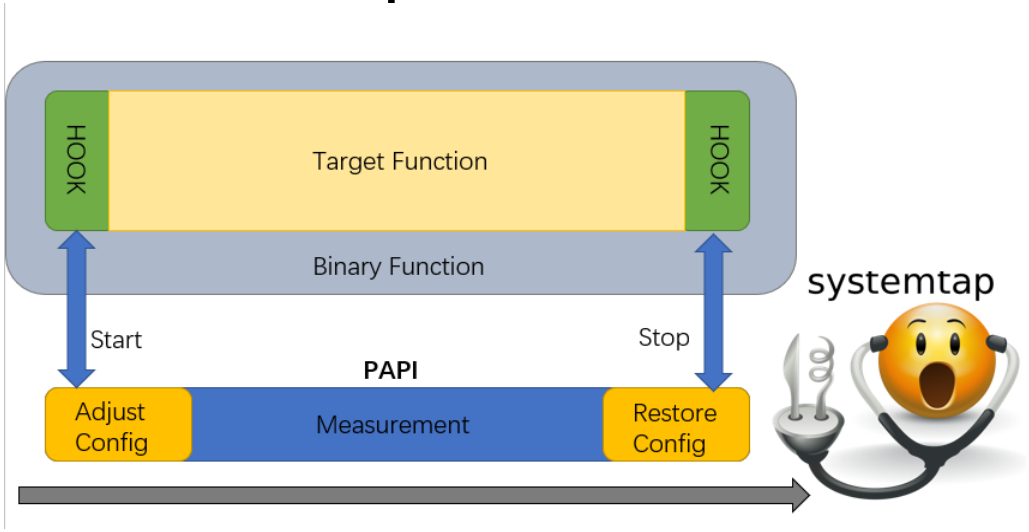
- The Helper Threads on the “unused cores” monitor the execution of user process and provide optimization processing.



Functions of the UTHelper are... (example)
①Real-time performance profiling
②Automatic adjustment of the number of parallelism and thread affinity

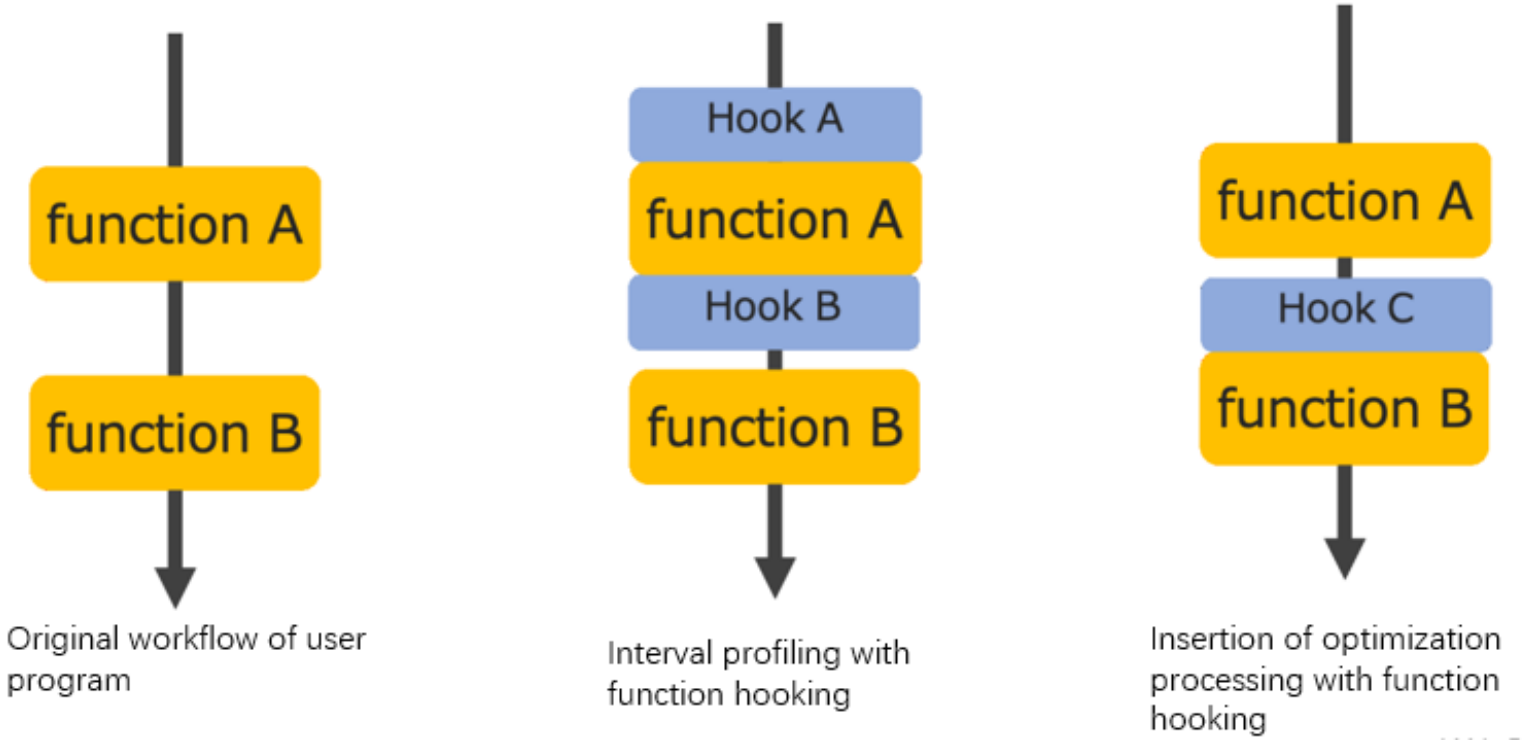
Implementation

- SystemTap[1] detects the occurrence of events, runs the handler as a quick sub-routine and then resumes.
- PAPI[2] access the hardware performance counters and collect performance metrics.



[1] Eigler, Frank Ch, and Red Hat. "Problem solving with systemtap." Proc. of the Ottawa Linux Symposium. Citeseer, 2006.
[2] Mucci, Philip J., et al. "PAPI: A portable interface to hardware performance counters." Proceedings of the department of defense HPCMP users group conference. Vol. 710. 1999.

Function hooking



Interval profiling: **PAPI, PAPI-C**[3]

Optimization processing: **OMPT**[4]

[3] Terpstra, D., Jagode, H., You, H., Dongarra, J. **"Collecting Performance Data with PAPI-C“**
[4] Eichenberger, Alexandre E., et al. "OMPT: An OpenMP tools application programming interface for performance analysis." International Workshop on OpenMP. Springer, Berlin, Heidelberg, 2013.

Preliminary experiments

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

- Dynamically collecting performance information like clock cycles during the execution of NAS Parallel Benchmark FT
- Evaluating the negative impact to main computation when running heavy computation on the “unused cores”

Results

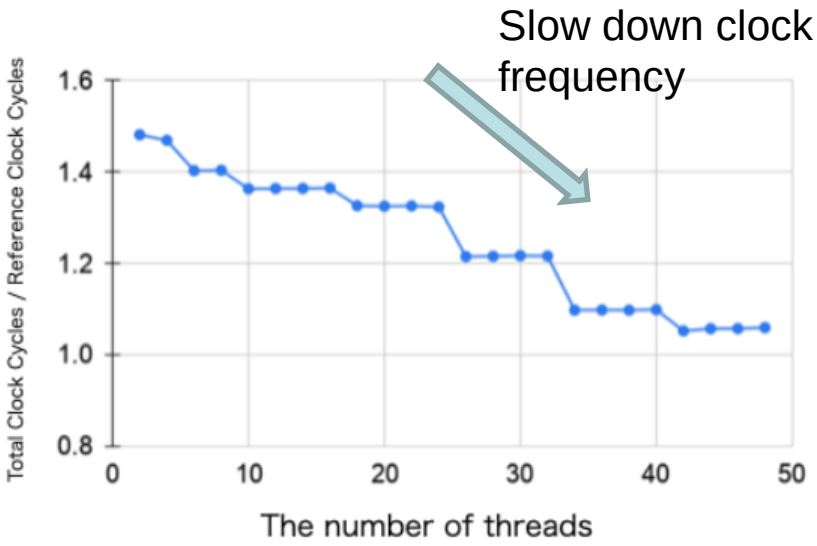


Fig a. Ratio of total clock cycles over reference clock cycle during execution

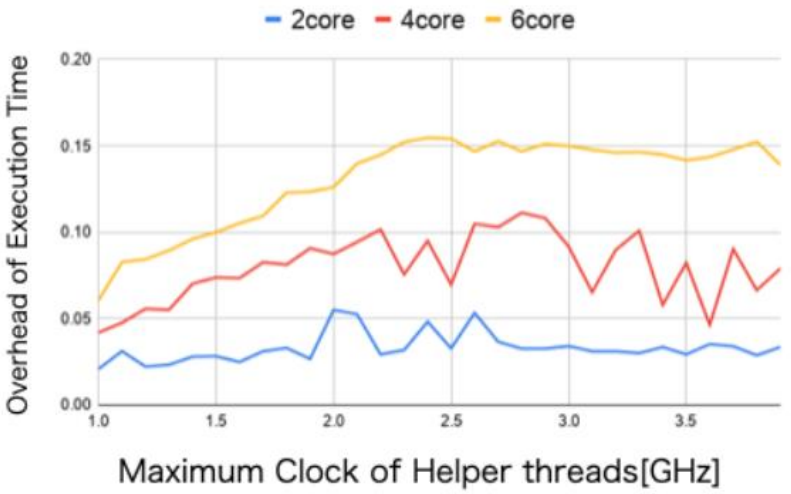


Fig b. Overhead of execution time over the baseline

#threads ↑, computing clock cycles ↓, which indicates the degradation of Turbo Boost.

#(dark cores) ↑, overhead of execution time ↑

Conclusion and future work

- Dynamic profiling on the unused can be achieved.
- The overhead to main computation is not significant.
- Optimization processing like dynamic adjusting the number of threads and thread affinity based on the result of profiling will be researched in the future.

Reference

[1] Eigler, Frank Ch, and Red Hat. "Problem solving with systemtap." Proc. of the Ottawa Linux Symposium. Citeseer, 2006.

[2] Mucci, Philip J., et al. "PAPI: A portable interface to hardware performance counters." Proceedings of the department of defense HPCMP users group conference. Vol. 710. 1999.

[3] Terpstra, D., Jagode, H., You, H., Dongarra, J. "Collecting Performance Data with PAPI-C“

[4] Eichenberger, Alexandre E., et al. "OMPT: An OpenMP tools application programming interface for performance analysis." International Workshop on OpenMP. Springer, Berlin, Heidelberg, 2013.

Thank you for Listening!!