Application Characterization by using Hardware Performance Counters with Data Mining

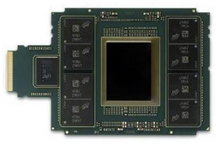
Jieun Choi, Geunchul Park, and Dukyun Nam

Supercomputing Center at the Korea Institute of Science and Technology Information (KISTI)

Introduction

Application-awareness in Supercomputer

- Basically, the minimum resources required by the application must be guaranteed.
- With the advent of processors including new architecture, application-awareness in HPC become difficult.
  - Additional on-package & high-bandwidth memory
  - More scalable processor or Self-booting processor
  - High-speed interconnect architecture etc.



- Conventional analysis approaches such as code static analysis or run-time monitoring are *not* sufficient

Proposed Application Characterization Method

- ✓ **Trace and profile processor behavior** using hardware performance counter
- ✓ **Simply and quickly classify** application characteristics with data mining
- ✓ **Provide insights** for resource management and performance optimization

Backgrounds & Related Works

Hardware Performance Counters

- Built into the microprocessor as a set of special purpose registers
- Count the number of occurrences of performance-related hardware events
- Can profile and trace performance events using Linux Perf

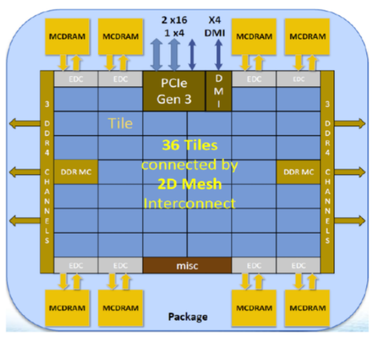
Application Profiling Tools

Tool (Institute)	Description
Vtune (Intel)	• Use the HPC analysis using three critical metrics(CPU utilization, memory access, FPU utilization) • Supply a single analysis interface for all the performance analysis on modern processors
TAU (Univ. of Oregon)	• Instrument source code for routines, loops, and memory • Measure time spent in each routine by profiling and tracing hardware performance counter • Can utilize diverse visualization tools for graphical displays of all the performance analysis results
Allinea (Arm)	• Highlight the slowest applications at a glance • Provide a single page report about the computation, communication, and I/O activities

Intel Knights Landing Processor (KNL)

Hardware Specification

- Stand-alone processor (up to 72 cores)
  - 288 cores with hyper-threading
- Multi-channel DRAM (MCDRAM)
  - DDR4 : ~90GB/s (up to 384GB)
  - MCDRAM: ~465GB/s (up to 16GB)
- 3 possible HBM memory models
  - Cache / Flat / Hybrid modes



➢ We categorize hardware events of the KNL according to three critical criteria (CPU, Memory, Disk I/O)

Category	Detailed Event List
CPU	Instructions, Branch Instructions, Branch Misprediction, iTLB loads, iTLB load misses, L1 icache loads, L1 icache load misses, UOPS_RETIRED.ALL, UOPS_RETIRED.SCALAR_SIMD, MEM_UOPS_RETIRED.ALL_Stores
Memory	Cache misses, cache references, L1-dcache-load-misses, LLC-loads, dTLB-load-misses, MEM_UOPS_RETIRED.L2_HIT_LADS, MEM_UOPS_RETIRED.L2_MISS_LOADS, L2_REQUESTS_REJECT.ALL, L2_REQUESTS_REJECT_REFERENCE, L2_REQUESTS_REJECT_MISS
Disk I/O	Scsi_dispatch_cmd_done, xfs_file_buffered_write, xfs_file_Read, xfs_get_block_allc, xfs_get_blocks_found

NAS Parallel Benchmark

- NPB is benchmark test programs for Computational Fluid Dynamics

APP.	Problem description	App. Characteristics
IS	Integer sort, random memory access	Compute
EP	Embarrassingly parallel, no communication between each process	Compute
CG	Conjugate gradient, irregular memory access	Memory
MG	Multi-grid on a sequence of meshes, Long- and short-distance communication	Compute & Memory
FT	Discrete 3D fast Fourier transform, all-to-all communication	Compute
BT	Block tri-diagonal solver	Compute
SP	Scalar penta-diagonal solver	Compute
LU	Lower-upper Gauss-Seidel solver	I/O
BT_epio	Each participating process writes data	I/O
BT_full	MPI I/O with collective buffering	I/O
BT_simple	MPI I/O without collective buffering	I/O

Application Characterization with Data Mining

Our Approach

- Overall Steps: Event Collection – Profiling – Application Characterization

Collect hardware performance events according to computing resource

[Event Collection Step]

- Record the number of hardware events occurrences with NAS Parallel Benchmarks (NPB) in 4 KNL nodes.
  - Intel® Xeon Phi™ CPU 7250 @ 1.40GHz, 68cores (enable hyper-threading)
  - 96GB DDR4 and 16GB MCDRAM memory (cache mode)

Generate refined information by profiling raw events with data mining

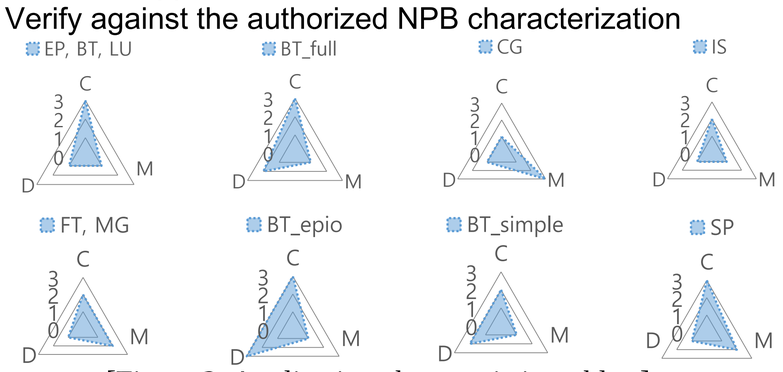
[Profiling Step]

- Process the collected data by applying Expectation-Maximization (EM) clustering technique using Weka tools.
  - EM clustering is representative probability-based algorithm based on the probability for inclusion in clusters.
  - Execute EM clustering 3 times for each event category.
  - Set the value of cluster parameter to 3 which means relative resource usage

Gain insights and draw tables/charts for application characteristics

[Application Characterization Step]

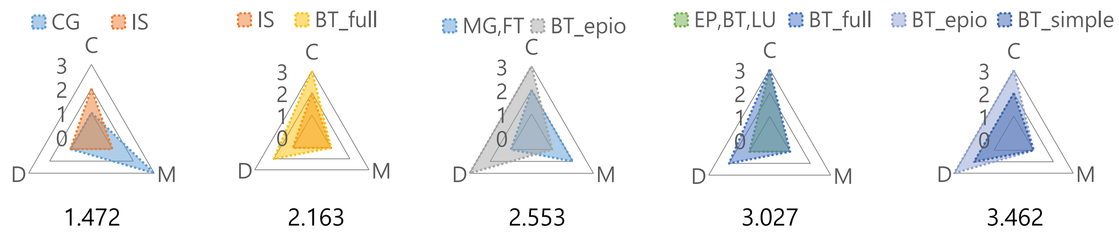
- Generate application characterization tables
- Verify against the authorized NPB characterization



[Figure 2: Application characteristics tables]

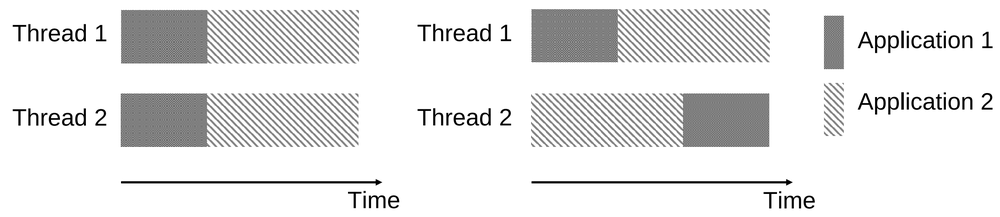
Case Study: Interference Analysis with Application Characteristics

- We assumed that a larger overlapping area of the charts indicates greater contention and interference between resources.



[Figure 2: Overlapping area through two application characteristics table]

- Experimental scenarios



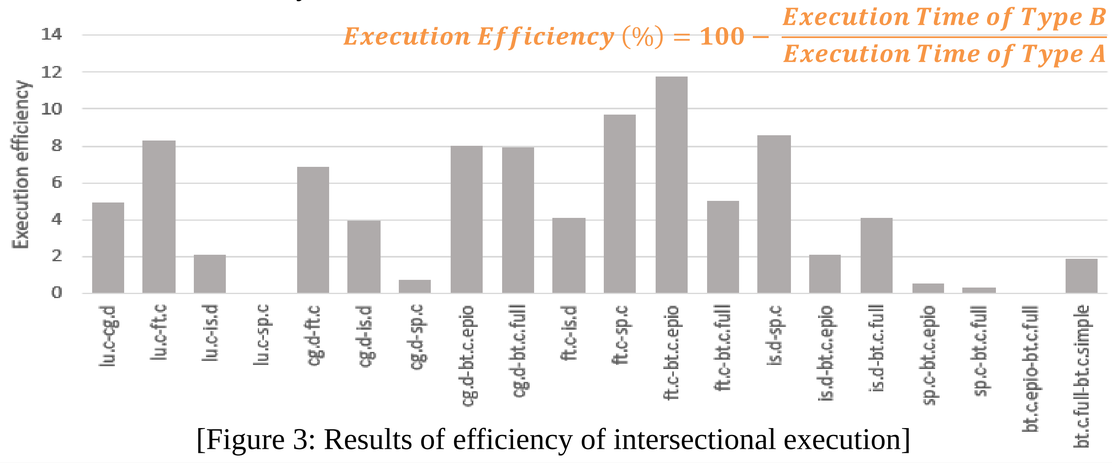
[Sequential Execution: the second application was performed after each thread executed the first application]

[Intersectional Execution: the two threads performed different applications, and they then changed applications]

- Experimental results
  - Pearson correlation coefficient : -0.374 (the case of seven I/O-intensive cases were excluded)

  ⇒ There is a significant negative linear correlation between interference ratios execution efficiency.

$$Execution\ Efficiency\ (\%) = 100 - \frac{Execution\ Time\ of\ Type\ B}{Execution\ Time\ of\ Type\ A}$$



[Figure 3: Results of efficiency of intersectional execution]

Conclusion & Future Work

- In this paper, we have laid the foundation for the whole performance profiling software development in supercomputer.
- We will continue to investigate inter-node communication patterns in cluster system.
- We plan to additional performance optimization experiments for disk I/O-intensive application.