Online regulation of power draw for energy savings in HPC kernels

Piotr Luszczek

Scheduling Variable Capacity Resources for Sustainability
University of Chicago, Paris, March 30, 2023

Anthony Danalis, UTK

Heike Jagode, UTK

Daniel Barry, UTK

Johannes Gebert, HLRS
Motivation:

Microprocessors now have enough functionality in hardware to be energy optimal beyond what software can do.

Anonymous Intel Hardware Developer, ~2015

Based on models embedded in x86 silicon?
- Data prefetcher
- Branch predictor
- Power consumption model based on performance counters
- Overheating detection
  - Temperature sensors
  - Voltage regulators
  - Separate frequency multipliers for different units
“Last Mile” Power Draw Reading and Control for Users

- ACPI-P states
  - Frequency and voltage scaling
- ACPI-C states
  - Functional unit control including extended states for stopping mid-instruction
- Thermal sensors on POWER6
- Intel RAPL in Sandy Bridge
- PAPI Power counters
  - ENERGY_UJ:ZONE0
  - ENERGY_UJ:ZONE0_SUBZONE
  - POWER_LIMIT_A_UW:ZONE0
- Kernel setting
  echo 0 > /proc/sys/kernel/perf_event_paranoid
Power-Capping DGEMV(18k) on KNL (flat mode)

- DDR4 memory
- MCD memory
Data Cleaning and Event Counters Selection

• Data cleaning
  • Counter data is precise but almost never exactly repeatable
  • System noise always present
  • Classic methods make assumptions error distribution and bias
    • SVD, QRCP, isotonic regression,
  • More complex measures required for capturing non-linear effects needs
    • Entropy, Genie information impurity

• Event counter selection
  • Using more counters provides better kernel characterization
  • Number of events monitored simultaneously has always been small
  • Multiplexing has limits and degrades accuracy
  • Applications need counters (but we’re not ready to share counters yet)
Power Profiles for XSbench (MC Neutron Transport)

- 160W (-3.4%)
- 140W (0%)
- 120W (10%)
- 100W (20%)
- 90W (24%)
- 80W (16%)

Intel Cascade Lake

<table>
<thead>
<tr>
<th>Power Cap</th>
<th>Energy Savings</th>
</tr>
</thead>
<tbody>
<tr>
<td>200 W</td>
<td>-3.4 %</td>
</tr>
<tr>
<td>160 W</td>
<td>0.1 %</td>
</tr>
<tr>
<td>140 W</td>
<td>0.1 %</td>
</tr>
<tr>
<td>120 W</td>
<td>10.1 %</td>
</tr>
<tr>
<td>100 W</td>
<td>19.8 %</td>
</tr>
<tr>
<td>90 W</td>
<td>23.8 %</td>
</tr>
<tr>
<td>80 W</td>
<td>15.9 %</td>
</tr>
</tbody>
</table>
Data Model Selection (Digital Mini-Twin)

• Analytic performance models focus on performance metrics rather than operation of individual functional units that might be throttled
  • But may still be used for automated labelling

• Offline data models assume exclusive use of the machine
  • Interference with the monitored application is not considered

• Assigning labels to compute kernels needs automation
  • Kernel behavior depends on the compiler, flags, system state (cold/hot cache)
  • Using two labels (compute- or memory-bound) was not sufficient for power capping

• Mixture of models improves prediction quality and inference overhead
  • Reusing existing models allows for greater range of choices to minimize overhead
    • Changing hardware counters may force a change of model
  • Many time series models to choose from
    • sktime, greykite, prophet, timemachines

• Decision trees achieve the best overhead-accuracy trade-off
Powercap’s Influence on Time and Energy

Region of optimal energy savings in memory-bound states.
Software and API Design

• Simple interface to mimic existing calls for counter and events
  • `start()` and `stop()` calls for marking code regions
    • `event_start()` and `event_stop()` for counter reading
    • `powercap_start()` and `powercap_stop()` for powercapping

• Compatibility with performance tools enables their automation of inserting the start/stop calls around kernels of interest

• Online model runs in a separate thread and needs a single core for inference

• The models come in as weights evaluated with (almost always) Python code
  • Lowering Python code down to typed and compiled code limits flexibility
  • Minimizing overhead of bytecode execution is not essential (yet) because of `numpy`

• Sampling frequency must balance accuracy, responsiveness, inference overhead, kernel duration, and hardware support
Online Power ML Model for Power-Capping: XSBench
Online Power ML Model for Power-Capping: Jacobi

Energy Savings: 9.1 %
Time Cost: -2.3 %

Intel Cascade Lake
Future Directions

• Model improvements
• Accounting for hardware capacity and/or density
• Heterogenous environments