Slow Down or Halt: Saving the Optimal Energy for Scalable HPC Systems

Li Tan and Zizhong Chen

University of California, Riverside

ICPE’15, Work-in-Progress Paper, Austin, TX, USA
February 3, 2015
Power/Energy Concerns in HPC

- Power and energy consumption of High Performance Computing (HPC) is a growing severity → operating costs and system reliability.
  - Launching date of exascale computers is approaching
  - Slack is pervasive during HPC runs

- Dynamic Voltage and Frequency Scaling (DVFS)
  - voltage/frequency ↓ → power ↓ → energy efficiency
  - Strategically switch processors to low-power states when the peak processor performance is unnecessary
    - Slack: hardware waiting periods from imbalanced workload
    - Communication delay, load imbalance, memory access etc.
Effectiveness of DVFS Approaches

› Basics of DVFS
  › A runtime technique that is able to switch *operating frequency* and *supply voltage* of *power-scalable* hardware components (CPU, GPU, memory, etc.) to different *levels* per *workload characteristics* to energy↓

› In this Work ➔ Two DVFS Approach Comparison
  › Critical Path aware Slack Reclamation
  › Race-to-halt (also known as Race-to-idle)
  › Theoretical + experimental head-to-head comparison
Two Classic Energy Saving Solutions
Theoretical Energy Savings

Strategy I (Race-to-halt): Execute $t$ at the highest frequency $f_h$ until the end, and then switch to the lowest frequency $f_l$, i.e., run in $T$ at $f_h$ and in $T'$ at $f_l$;

Strategy II (CP-aware Slack Reclamation): Execute $t$ at the optimal frequency $f_m$ with which $T'$ is eliminated, i.e., run in $T + T'$ at $f_m$ (For simplicity in the later discussion, assume $T'$ can be eliminated using available frequency $f_m$ without frequency approximation).

Table 2: Frequency-Voltage Pairs for Different Processors (Unit: Frequency (GHz); Voltage (V)).

<table>
<thead>
<tr>
<th>Gear</th>
<th>AMD Opteron 2380</th>
<th>AMD Opteron 846 and AMD Athlon64 3200+</th>
<th>AMD Opteron 2218</th>
<th>Intel Pentium M</th>
<th>Intel Pentium 4 HT 530</th>
<th>Intel Xeon E5 2687W</th>
<th>Intel Core i7-2760QM</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2.5</td>
<td>1.300</td>
<td>2.0</td>
<td>1.500</td>
<td>2.4</td>
<td>1.250</td>
<td>1.4</td>
</tr>
<tr>
<td>1</td>
<td>1.8</td>
<td>1.200</td>
<td>1.8</td>
<td>1.400</td>
<td>2.2</td>
<td>1.200</td>
<td>1.2</td>
</tr>
<tr>
<td>2</td>
<td>1.3</td>
<td>1.100</td>
<td>1.6</td>
<td>1.300</td>
<td>1.8</td>
<td>1.150</td>
<td>1.0</td>
</tr>
<tr>
<td>3</td>
<td>0.8</td>
<td>1.025</td>
<td>0.8</td>
<td>0.900</td>
<td>1.0</td>
<td>1.100</td>
<td>0.8</td>
</tr>
</tbody>
</table>
Theoretical Energy Savings (Cont.)

\[ P = P_{\text{dynamic}}^{CPU} + P_{\text{leakage}}^{CPU} + P_{\text{leakage}}^{\text{other}} \]  
(1)

\[ P_{\text{dynamic}} = ACfV^2 \]  
(2)

\[ P_{\text{leakage}} = I_{\text{sub}}V \]  
(3)

\[ E(S_1) = \overline{P(S_1)} \times T + \overline{P'(S_1)} \times T' \]
\[ = (ACf_hV_h^2 + I_{\text{sub}}V_h + P_c)T + (ACf_iV_i^2 + I_{\text{sub}}V_i + P_c)T' \]
\[ = AC(f_hV_h^2T + f_iV_i^2T') + I_{\text{sub}}(V_hT + V_iT') + P_c(T + T') \]  
(5)

\[ E(S_2) = \overline{P(S_2)} \times (T + T') \]
\[ = (ACf_mV_m^2 + I_{\text{sub}}V_m + P_c)(T + T') \]
\[ = ACf_mV_m^2(T + T') + I_{\text{sub}}V_m(T + T') + P_c(T + T') \]  
(6)

\[ E(S_2) - E(S_1) = AC \left( (f_mV_m^2 - f_hV_h^2)T + (f_mV_m^2 - f_iV_i^2)T' \right) \]
\[ + I_{\text{sub}} \left( (V_m - V_h)T + (V_m - V_i)T' \right) \]  
(7)
Experimental Power Savings

Matrix Size: 160000 x 160000, Power Costs of Three Nodes

Figure 2: Power Costs of Cholesky Factorization with Two Energy Saving Solutions on Cluster ARC.
Figure 3: Performance of Cholesky Factorization with Two Energy Saving Solutions on Cluster ARC.
Conclusions and Future Work

- Energy saving gap between two classic solutions is **narrowed down** for current HPC systems
  - CP-aware Slack Reclamation vs. Race-to-halt
  - State-of-the-art CMOS technologies allow insignificant variation of supply voltage as operating frequency of a processor scales up and down ➔ **voltage scales much less than frequency and the trend continues!**

- Ongoing Directions
  - Model and generalize both solutions for more app.
  - Apply improved solutions on emerging architectures