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

UCR
UNIVERSITY OF CALIFORNIA, RIVERSIDE

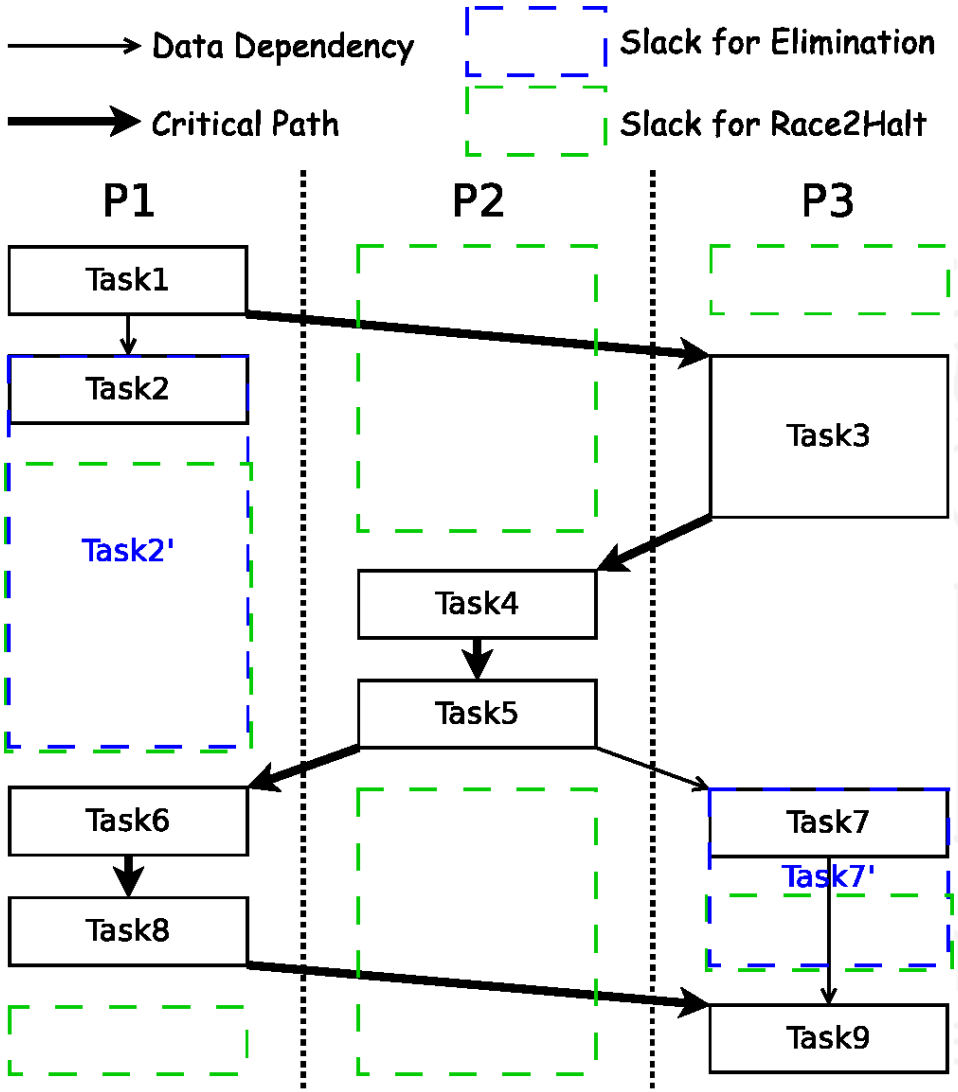
Power/Energy Concerns in HPC

- 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)).

Gear	AMD Opteron 2380		AMD Opteron 846 and AMD Athlon64 3200+		AMD Opteron 2218		Intel Pentium M		Intel Pentium 4 HT 530		Intel Xeon E5 2687W		Intel Core i7-2760QM	
	Freq.	Volt.	Freq.	Volt.	Freq.	Volt.	Freq.	Volt.	Freq.	Volt.	Freq.	Volt.	Freq.	Volt.
0	2.5	1.300	2.0	1.500	2.4	1.250	1.4	1.484	3.0	1.430	3.1	1.200	2.4	1.060
1	1.8	1.200	1.8	1.400	2.2	1.200	1.2	1.436	N/A	N/A	N/A	N/A	2.0	0.970
2	1.3	1.100	1.6	1.300	1.8	1.150	1.0	1.308	N/A	N/A	N/A	N/A	1.6	0.890
3	0.8	1.025	0.8	0.900	1.0	1.100	0.8	1.180	2.1	1.250	1.2	0.840	0.8	0.760

Theoretical Energy Savings (Cont.)

$$P = P_{dynamic}^{CPU} + P_{leakage}^{CPU} + P_{leakage}^{other} \quad (1)$$

$$P_{dynamic} = ACfV^2 \quad (2)$$

$$P_{leakage} = I_{sub}V \quad (3)$$

$$\begin{aligned} E(S_1) &= \overline{P(S_1)} \times T + \overline{P'(S_1)} \times T' \\ &= (ACf_hV_h^2 + I_{sub}V_h + P_c)T + (ACf_lV_l^2 + I_{sub}V_l + P_c)T' \\ &= AC(f_hV_h^2T + f_lV_l^2T') + I_{sub}(V_hT + V_lT') + P_c(T + T') \end{aligned} \quad (5)$$

$$\begin{aligned} E(S_2) &= \overline{P(S_2)} \times (T + T') \\ &= (ACf_mV_m^2 + I_{sub}V_m + P_c)(T + T') \\ &= ACf_mV_m^2(T + T') + I_{sub}V_m(T + T') + P_c(T + T') \end{aligned} \quad (6)$$

$$\begin{aligned} E(S_2) - E(S_1) = AC\left((f_mV_m^2 - f_hV_h^2)T + (f_mV_m^2 - f_lV_l^2)T'\right) \\ + I_{sub}\left((V_m - V_h)T + (V_m - V_l)T'\right) \end{aligned} \quad (7)$$

Experimental Power Savings

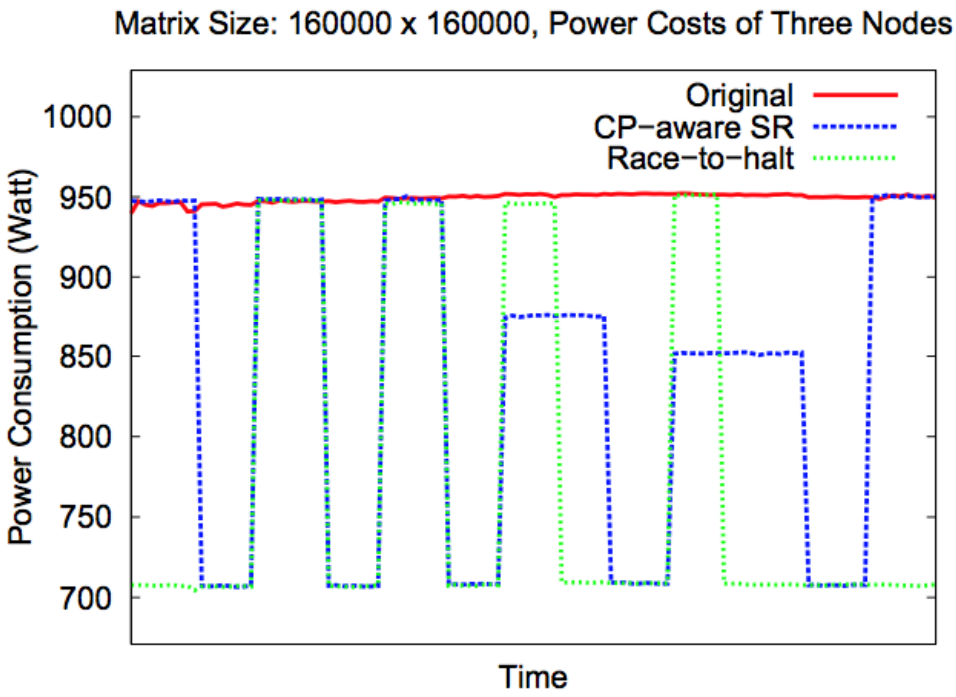


Figure 2: Power Costs of Cholesky Factorization with Two Energy Saving Solutions on Cluster ARC.

Experimental Performance Loss

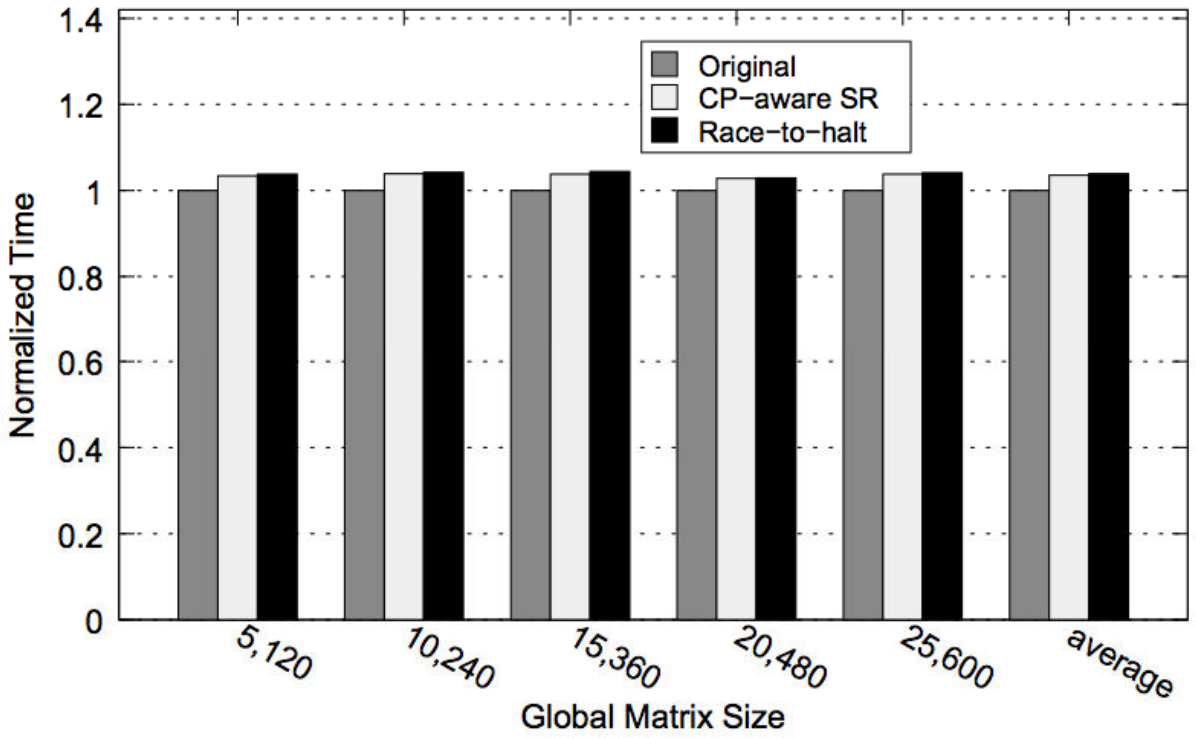


Figure 3: Performance of Cholesky Factorization with Two Energy Saving Solutions on Cluster ARC.

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