Predicting Performance Impact of DVFS for Realistic Memory Systems

Rustam Miftakhutdinov
The University of Texas at Austin
rustam@hps.utexas.edu

Eiman Ebrahimi
Nvidia Corporation
ebrahimi@hps.utexas.edu

Yale N. Patt
The University of Texas at Austin
patt@hps.utexas.edu

DVFS Performance Prediction for Energy Efficiency

Dynamic voltage and frequency scaling (DVFS) can make modern processors more energy efficient if we can accurately predict the effect of frequency scaling on performance. With DVFS support, a processor can alter its performance and power consumption on the fly by changing its frequency (and adjusting voltage accordingly). If the processor can accurately predict what its performance and power consumption would be at any frequency, it can switch to the frequency that minimizes overall energy consumption.

DVFS Control Overview

1. While running at frequency f_o, measure workload parameters for an interval
2. Based on measured workload parameters, predict execution time and power consumption as functions of frequency f, as shown in Figure 1

> ! **Our goal is to provide an accurate performance predictor for this step.**

3. Compute energy(f) = time(f) × power(f)
4. Find f_{opt} where energy(f) is minimal (Figure 2)
5. Switch frequency to f_{opt} for the next interval

Time/Power vs. Frequency

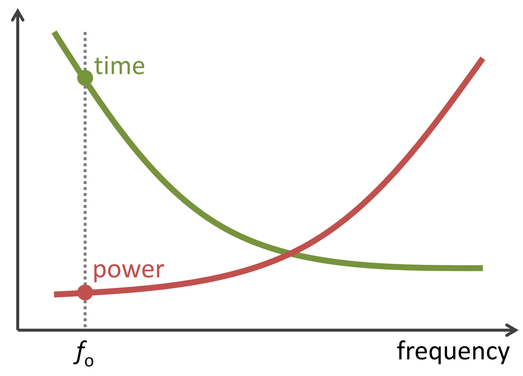

Figure 1. Time and power as functions of frequency. Both are predicted by DVFS controller.

Energy vs. Frequency

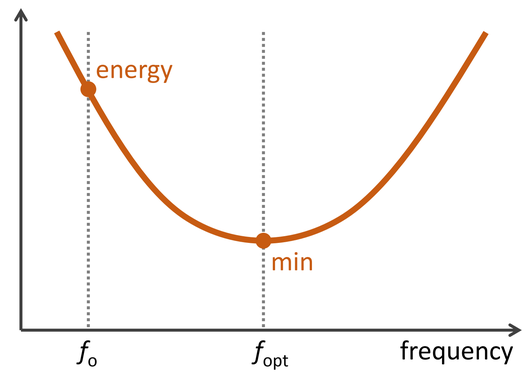

Figure 2. Energy as function of frequency. Ideally, the chip runs at f_{opt} where energy is minimal.

Realistic Memory Systems in Design and Evaluation

DVFS performance predictors must be evaluated with realistic memory systems, because simplified evaluation leads to wrong conclusions. Figure 3 shows normalized energy saved by DVFS using:

- Offline optimal DVFS policy (*potential*)
- Prior work (*stall time* and *leading loads*)
- Our DVFS performance predictor (*our predictor*).

Note that prior work, originally evaluated with a simplified constant access latency memory system, fails when confronted with realistic memory systems. Our predictor, designed for realistic memory systems, maintains high energy savings throughout.

Results

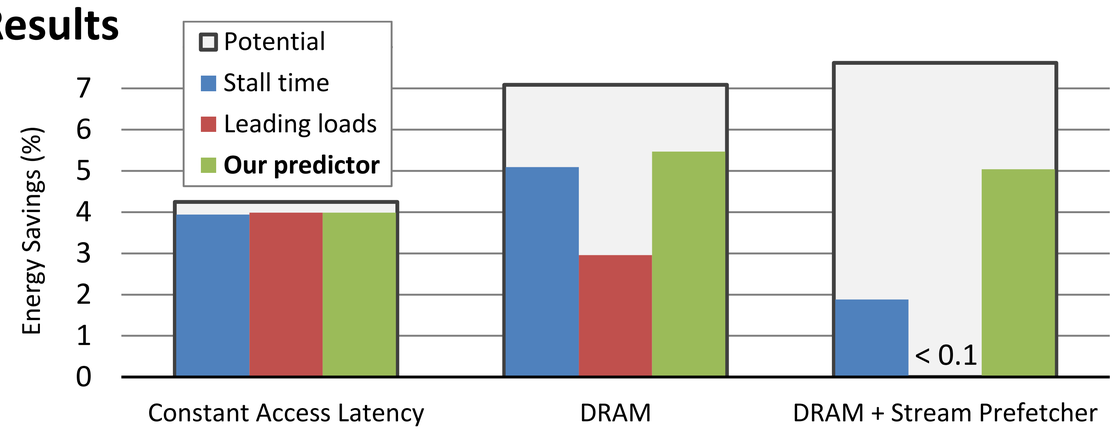

Figure 3. Potential and realized normalized energy savings. Geometric mean over 13 memory-intensive SPEC 2006 benchmarks. Baseline: most energy efficient static frequency for SPEC 2006.

System Configuration

DVFS	1.5–4.5 GHz, 100K inst. interval
Front end	4 wide (2 BRs/cycle), 4K BTB entries, Hybrid BP (64K gshare+64K PAs)
OOO core	4 wide, 14 stages, ROB: 128, RS: 48
Caches	64B line, 32 MSHRs. Inst: 32KB (3 c.), Data: 32KB (3 c.), L2: 1MB (18 c.)
DRAM	DDR3, 800MHz bus, 8 chips × 256 MB, 8 banks, 8KB row, CAS = 13.75 ns
MC	FR-FCFS, 32 request window
Stream prefetcher	64 streams, distance: 64, degree: 4, Queue size: 128

Handling Variable Access Latency

DVFS performance predictors must account for variable access latency, a key characteristic of realistic memory systems. Memory access latencies vary depending on whether the accesses conflict in DRAM banks and row buffers.

Our DVFS performance predictor handles variable access latency by estimating T_{memory} (a parameter of the linear DVFS performance model shown in Figure 4) as the length of the critical path through the memory requests (Figure 5).

Linear Model

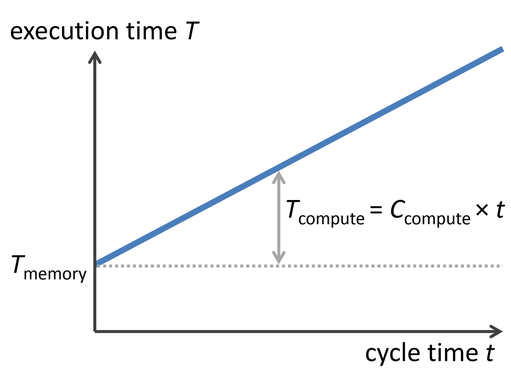

Figure 4. Linear DVFS performance model introduced by prior work.

Execution Example

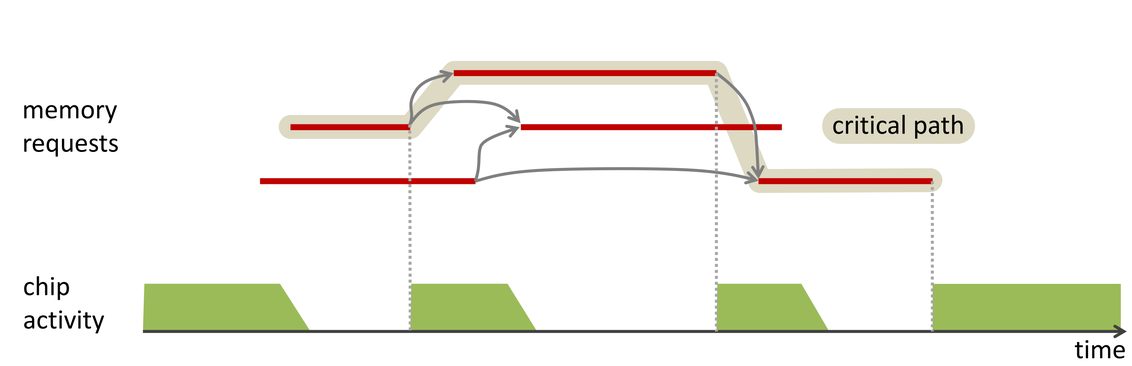

Figure 5. Out-of-order processor execution with a variable access latency memory system. Gray arrows denote memory request dependences. We estimate T_{memory} as length of the highlighted critical path.

Handling DRAM Bandwidth Saturation due to Prefetching

DVFS performance predictors must also account for prefetching, a common feature of modern chips.

While prefetching does reduce memory-related processor stalls (left side of Figure 6), prefetching also exposes a new performance limiter: DRAM bandwidth (right side of Figure 6).

To handle DRAM bandwidth saturation, we introduce the limited bandwidth DVFS performance model (Figure 7) and design hardware mechanisms to measure its parameters.

Execution Example

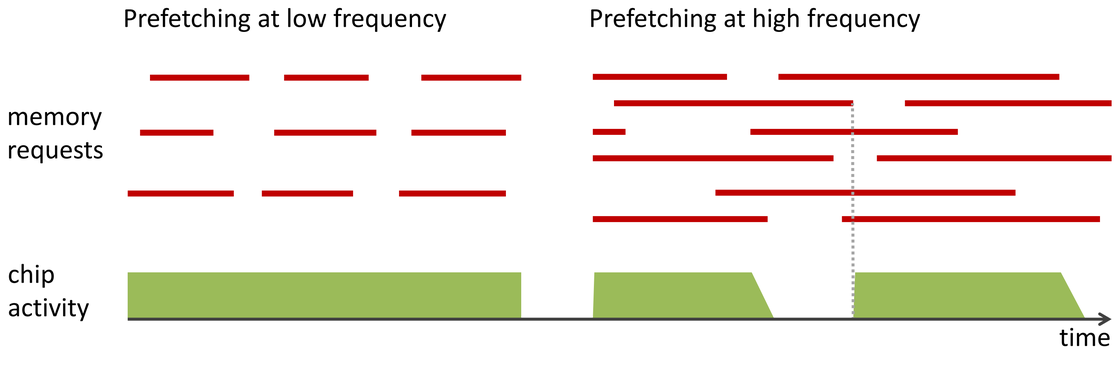

Figure 6. Out-of-order processor execution with a stream prefetcher. Note that, at high frequencies, the processor may stall on prefetch requests due to DRAM bandwidth saturation.

Limited Bandwidth Model

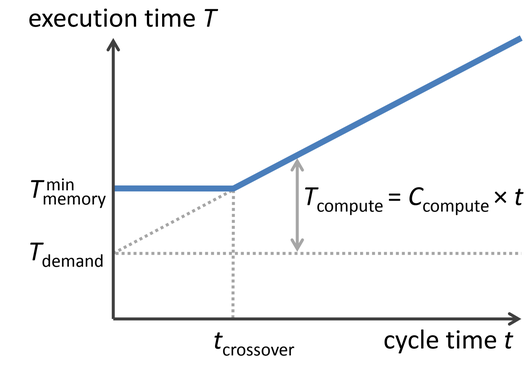

Figure 7. Limited bandwidth DVFS performance model that accounts for DRAM saturation.