Voltage Regulator Efficiency Aware Power Management

Yuxin Bai
University of Rochester
abayax@gmail.com

Victor W. Lee
Intel Corporation
victor.w.lee@intel.com

Engin Ipek
University of Rochester
ipek@cs.rochester.edu

Abstract

Conventional off-chip voltage regulators are typically bulky and slow, and are inefficient at exploiting system and workload variability using Dynamic Voltage and Frequency Scaling (DVFS). On-die integration of voltage regulators has the potential to increase the energy efficiency of computer systems by enabling power control at a fine granularity in both space and time. The energy conversion efficiency of on-chip regulators, however, is typically much lower than off-chip regulators, which results in significant energy losses. Fine-grained power control and high voltage regulator efficiency are difficult to achieve simultaneously, with either emerging on-chip or conventional off-chip regulators.

A voltage conversion framework that relies on a hierarchy of off-chip switching regulators and on-chip linear regulators is proposed to enable fine-grained power control with a regulator efficiency greater than 90%. A DVFS control policy that is based on a reinforcement learning (RL) approach is developed to exploit the proposed framework. Per-core RL agents learn and improve their control policies independently, while retaining the ability to coordinate their actions to accomplish system level power management objectives. When evaluated on a mix of 14 parallel and 13 multiprogrammed workloads, the proposed voltage conversion framework achieves 18% greater energy efficiency than a conventional framework that uses on-chip switching regulators. Moreover, when the RL based DVFS control policy is used to control the proposed voltage conversion framework, the system achieves a 21% higher energy efficiency over a baseline oracle policy with coarse-grained power control capability.

CCS Concepts • **Computer systems organization → Architectures**; • **Hardware → Power and energy**

ASPLOS '17, April 08-12, 2017, Xi'an, China
© 2017 ACM. ISBN 978-1-4503-4465-4/17/04...$15.00
DOI: http://dx.doi.org/10.1145/3037697.3037717

Keywords Computer architecture; Power, energy, and thermal management.

1. Introduction

Energy efficiency is a critical requirement in computer systems, from mobile platforms to desktop computers and enterprise servers. In a mobile device, the energy consumption must be kept low to sustain a sufficient battery lifetime; at a data center, the energy usage directly affects the recurring costs of ownership. Power management plays a significant role in achieving high energy efficiency. One of the most effective power management techniques, *Dynamic Voltage Frequency Scaling (DVFS)*, involves adaptively adjusting the supply voltage and clock frequency at runtime to reduce the power dissipation. Modern processors are equipped with DVFS capability to permit switching between power states, and there is substantial work in designing DVFS power management policies under the assumption of a lossless voltage conversion framework [12, 22, 44, 47, 50].

One of the key components that affects the efficiency of DVFS power management is a *Voltage Regulator (VR)*. A VR converts the input voltage from the noisy power supply into one or more desired voltage levels to be used by the processor. Recent industry trends are toward integrating voltage regulators on chip [18], such that the granularity of DVFS in space (more voltage domains), time (faster response time), and voltage levels (smaller voltage steps) can all be improved, thereby enabling greater opportunities for more efficient power management. The energy conversion efficiency of most on-chip regulators, however, are significantly lower than off-chip voltage regulators that can only support coarse-grained power control. For example, commonly used voltage regulators, such as switching regulators and switched-capacitor regulators, suffer from significant energy efficiency losses (over 20% [23, 24, 31]). One type of linear regulator, the low dropout (LDO) voltage regulator, is generally considered suitable for on-chip integration [43] due to its area and speed advantages. The energy conversion efficiency of an LDO regulator, however, is dependent on the ratio of its output and input voltages; as a result, an LDO regulator can provide a high energy conversion efficiency within only a small voltage range, and is energy inefficient if it is used to supply a wide range of voltages. To achieve both fine-granularity and high regulator efficiency, it is therefore extremely important to be aware of the on-chip regulator efficiency losses, and to adapt the voltage conversion framework and power control policies accordingly.

Existing control policies for DVFS typically rely on feedback-based control theory, or system power and performance modeling to identify the optimal voltage and frequency (VF) level. Simple approaches are often inaccurate in reacting to complex system behaviors. Sophisticated models, either analytical or statistical, require prior knowledge and careful calibration for a specific system; consequently, these models are difficult to port to new systems or customize to a given workload. Moreover, existing control policies are unaware that their decisions may affect the underlying regulator efficiency, which makes these policies generally incapable of making optimal DVFS control decisions in the context of on-chip regulators. Therefore, a model-free control policy that is aware of the voltage conversion framework, that learns across various systems and workloads, and takes runtime uncertainties into account is desirable.

We observe that on-chip voltage regulator efficiency significantly affects the achievable energy efficiency of the system. We propose a voltage conversion framework for DVFS power management, in which off-chip switching regulators and on-chip LDO regulators are organized in a hierarchy to facilitate per-core nanosecond response and a high regulator efficiency. The proposed framework achieves 18% greater energy efficiency than a typical per-core framework using only on-chip switching regulators. In addition, we propose a new control policy for DVFS, which employs a reinforcement learning approach to evolve optimal policies for an arbitrary system or workload at runtime. The learned policy takes the on-chip regulator efficiency loss into account, and minimizes the energy under a parameterized performance constraint. This policy is 21% more energy efficient as compared to an oracle policy with only coarse-grained power control.

2. Background

In this section, three commonly used voltage regulators are introduced, and their suitability for on-chip integration is discussed. Related work on DVFS, and the application of reinforcement learning techniques to architectural control problems is reviewed.

2.1 Voltage Regulators

In a typical power management system, an AC voltage is first converted to a DC voltage, after which the output is regulated via DC-DC converters that generate the required voltages for the processor and other circuit blocks. A DC-DC converter is called a *voltage regulator*. Three primary functions of a voltage regulator are to 1) isolate the processor from fluctuations in the power supply, 2) convert the input voltage to the desired level, and 3) regulate the output voltage to the processor, so that the voltage is insensitive to variations in input voltage, load current, and temperature.

Linear Regulators. A linear regulator operates based on the principle of resistive voltage division. A low drop-out (LDO) regulator is a type of widely used linear regulator that allows the output to achieve a small drop-out voltage as compared to the input. Two advantages that make LDO regulators suitable for on-chip integration are 1) a small area due to the absence of passive inductors (capacitor free LDOs have also been proposed [8, 9, 32]), and 2) fast load voltage regulation with low noise. However, the limiting factor is the poor energy conversion efficiency due to the loss on the resistors in series and on the control logic. The efficiency is given by

$$\eta = \frac{V_{out}}{V_{in}}\eta_{current}, \tag{1}$$

where $\eta_{current}$ is the current efficiency. Modern LDOs achieve near-optimal current efficiency (more than 99% [29, 52]); consequently, the energy conversion efficiency is primarily determined by the V_{out}/V_{in} ratio. In other words, LDO regulators can achieve a high energy efficiency when the difference between the input and output voltages is kept small.

Switched Capacitor Regulators. A switched capacitor regulator, also referred as a charge pump converter, uses capacitors and switches to perform voltage conversion. Although lossless components are used, energy is consumed during the charge transfer. For on-chip integration, the switching frequency typically is increased to reduce the size of the capacitors, which results in greater dynamic power dissipation on the switches, thereby decreasing the energy conversion efficiency (typically less than 80% [15]). Another primary limitation is the poor regulation quality, which limits the application to scenarios where voltage regulation is not required.

Switching Regulators. Switching regulators, also referred as DC-DC or buck converters, are widely used between the battery and the microprocessor. To achieve the voltage regulation, a periodic signal is rectified and fed into an LC low pass filter to generate an output DC voltage. The level of this voltage is determined by the duty cycle of the periodic signal. To limit voltage and current ripples, the inductor and the capacitor must be sized up; consequently, switching regulators typically occupy a large area, and are employed off-chip. To realize on-chip integration, the switching frequency needs to be increased to reduce the size of the inductor and the capacitor. This increases the power loss, and significantly degrades the energy efficiency for on-chip integration scenarios (lower than 80% [25]).

The energy efficiency η for all three regulators are related to the input and output voltages, but the relationship is more complicated for switched capacitor and switching regulators due to the additional tradeoffs in area, speed, and regulation quality. Table 1 summarizes the characteristics of these three voltage regulators. **No existing regulator can enable both**

fine-grained power control (space and time) and high energy conversion efficiency at the same time. In this paper, a hybrid framework is proposed that uses both off-chip switching regulators and on-chip LDOs for power control. As a result, fine-grained power control (small area and fast response), and a high regulator energy conversion efficiency can be achieved simultaneously.

2.2 Dynamic Voltage and Frequency Scaling

As the supply voltage continues to shrink, the effectiveness of DVFS and other power management techniques such as power gating may be hampered. However, the impact of voltage regulator efficiency losses on the overall energy consumption still matters, and in fact becomes more prominent. For example, at low voltages, LDOs lose $\frac{1}{7}$ of the total energy when switching from $0.7V$ to $0.6V$, but only $\frac{1}{10}$ of the total energy when switching from $1.0V$ to $0.9V$.

Earlier papers on power management take advantage of DVFS in various ways. Su *et at.* [50] build a linear regression model that can predict performance and power based on hardware performance counters. Other researchers [12, 22, 44, 47] similarly rely on analytical models and hardware events to make predictions. Kim *et al.* [25] perform a sensitivity analysis on on-chip switching regulators, and point out that the slow response and the large area of off-chip regulators constitute obstacles to effective DVFS power management. Godycki *et al.* [15] and Sinkar *et al.* [46] identify the response time and area inefficiency of off-chip regulators, and propose switch-capacitor and LDO-based DVFS frameworks to save energy. Other DVFS works focus on management policies. A clustered mechanism is proposed by Yan *et al.* [59] to allow only one core within a cluster to use an on-chip regulator for fine-grained power control, while the rest of the cores can switch only in a coarse-grained fashion using off-chip regulators. Similarly to prior work by Rangan *et al.* [41], this approach requires frequently migrating the threads among the cores. Our work differs from all of these efforts in that 1) the on-chip regulator efficiency is addressed in the proposed framework, and 2) the RL-based control policy is aware of the regulator efficiency, and is portable across platforms without accurate modeling of a specific system.

2.3 Reinforcement Learning

Reinforcement learning [3] is a computational approach learning a goal via interaction with an environment. It has been successfully applied to a broad range of applications, such as autonomous navigation [48], robots [4, 26], games [3], network routing [7], scheduling [34], optimization [13], and resource allocation [37, 51]. In computer systems, reinforcement learning has also been successfully employed in memory controllers for optimally scheduling DRAM commands [21].

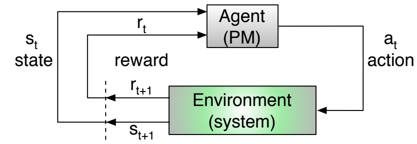


Figure 1. The agent-environment interaction in reinforcement learning.

As shown in Figure 1, a reinforcement learning system typically incorporates an *agent* (power manager) that acts as a learner and a decision maker, and an *environment* (system) that comprises everything outside the agent that interacts with the agent. These interactions occur in a sequence of discrete *steps* $t = 0, 1, 2, ...$; in each step, the agent selects an *action* (voltage-frequency pair) and applies it to the environment; the environment responds to the action and presents a new *state* back to the agent. The environment also returns a *reward* (energy consumption), a numerical value whose long term sum the agent tries to optimize. A *policy* specifies how the agent behaves, and is inherently a mapping from the environment states to the probabilities of selecting different possible actions. Reinforcement learning can be treated as a process in which the agent continually evolves the policy as a result of its experience.

Markov Decision Processes. In reinforcement learning, the dynamics between states and actions are described by a *Markov Decision Process (MDP)*, in which the next state and the expected reward value are determined by only the current state and action, instead of a complete history of all past states and actions. In the case of power management, the next system state and the energy reward depend on the current system state and the voltage-frequency pair selected; consequently, they can be approximated by an MDP and formulated within the reinforcement learning framework.

Goals and Rewards. An RL problem is formulated such that the cumulative sum of all rewards is the final target that the agent tries to optimize. The lifetime of the agent usually is infinite, and thus for practical purposes, a discounted cumulative reward is used as the objective function such that the sum of all future rewards converges in the limit:

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + ... = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \quad (2)$$

where γ, $0 \leq \gamma \leq 1$, is called the *discount rate*. γ determines the impact of future rewards, such that a reward received k steps in the future is worth only γ^k times as much as an immediately received reward. When $\gamma = 0$, the agent becomes “myopic” in that it is only concerned with the immediate reward. As γ approaches 1, the agent takes future rewards into account more strongly, and becomes more “farsighted”.

Policies and Action-Value Functions. Reinforcement learning is based on estimating value functions—*i.e.*, functions of state-action pairs that evaluate “how good” it is for the agent to take a certain action in a given state. Typically, the value function is defined with a policy π. The *action-value function for policy* π when taking action a in state s is evaluated at every time step t, and can be calculated using

$$Q^{\pi}(s, a) = E_{\pi}\{R_t | s_t = s, a_t = a\}. \quad (3)$$

Type of regulators	Area	Speed	Regulation	Efficiency	Applications
Linear regulator [27, 28, 53]	small $\sim$0.01mm^2	fast 1-100ns	good	limited by V_{out}/V_{in}	DRAM
Switched-capacitor regulator [23, 31]	medium on-chip: $\sim$0.2mm^2	medium on-chip: $\sim$100ns	poor	medium on-chip: low ¡80%	flash EEPROM
Switching regulator [24, 39, 49]	off-chip: large $\sim$10mm^2 on-chip: medium $\sim$1mm^2	off-chip: slow $\sim$100μs on-chip: medium $\sim$100ns	good	off-chip: high $\sim$95% on-chip: low ¡80%	hard disk, SRAM processors

Table 1. Comparison for three typical voltage regulators [43].

The policy π is a rule expressing how to select an action based on the value function. Typically, greedy policies are used in RL problems, such as the ϵ-*greedy* policy, in which the policy π selects an action in the current state with a maximum $Q^{\pi}(s,a)$ value, but is also permitted to pick a random action with a small probability of ϵ. This forces the agent to select an optimal action towards the objective most of the time, while also allowing the agent to explore the unknown regions of the state space.

Solution Methods. Temporal difference (TD) learning algorithms are one category of RL solution methods that are inherently on-line. TD methods enable learning in every time step, and convergence to an optimal policy is guaranteed if the number of steps is sufficiently large. This is particularly favorable for fine-grained power management with on-chip regulators. One of the TD methods, Q-learning, is employed in this work. The update rule in each time step for Q-learning is

$$Q(s,a) = (1-\alpha)\cdot Q(s,a) + \alpha[r + \gamma \cdot max_{a'} Q(s',a')] \quad (4)$$

in which the system state is transitioned from s to s' by taking action a. α is the learning rate that determines how quickly Q-values change in response to the updates, and facilitates $Q(s,a)$ values to converge. The term $max_{a'}Q(s',a')$ denotes the maximum Q value for the next state s' among all possible actions a'. The term $r + \gamma \cdot max_{a'}Q(s',a')$ represents the total reward, which includes the immediate reward r, plus the sum of all discounted future rewards.

3. Motivation

In this section, a novel analytical model showing the relationship between the system energy efficiency and the regulator efficiency is built. A workload analysis showing the potential gains from nanosecond DVFS using on-chip regulators is also presented.

3.1 Analytical Study of System Energy Efficiency With On-Chip Voltage Regulators

Energy loss due to on-chip regulators can consume a noticeable fraction of the overall system energy. When regulator efficiency is taken into account, the relationship between the supply voltage and system energy changes.

From Cubic to Quadratic. For static CMOS, the dynamic power consumption at frequency f and voltage V is given by $CV^2f\alpha$, where C is the load capacitance, and α is the activity factor. The frequency f is roughly linearly proportional to the voltage V; therefore, the total power is a cubic function of the voltage. When the LDO efficiency $\eta = \frac{V}{V_{in}}$ is taken into account, however, the total dynamic power consumption becomes,

$$Power_{dyn} = \frac{CV^2f\alpha}{\eta} = CVV_{in}f\alpha, \quad (5)$$

where V_{in} is the regulator input voltage, which comes from the battery or other sources. If we assume that V_{in} is fixed, the power becomes quadratically proportional to the voltage V, which makes it less effective to save power by scaling the voltage than typically assumed.

System Energy Efficiency and Voltage Regulator Efficiency. To model the system energy efficiency, analytical performance and power models are constructed. Based on the Roofline model [55], the performance of the system can be represented as:

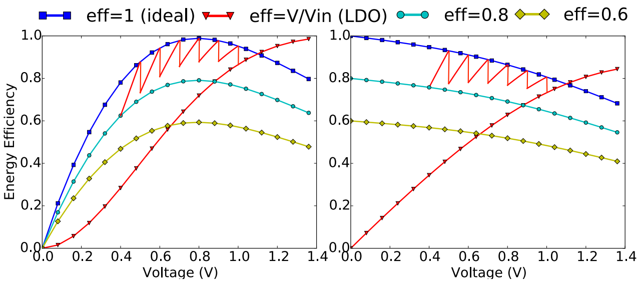
$$Perf. = \begin{cases} Flops_{peak} \times f \times \beta & (compute) \\ I \times BW & (mixed) \\ BW_{peak} \times I & (memory) \end{cases} \quad (6)$$

where the performance for compute-bound workloads is limited by the peak compute capability of the system in flops/sec (β is a parameter between 0 and 1), and is proportional to the frequency f. In contrast, the memory-bound workloads are constrained by the peak memory bandwidth of the system (bytes/sec). The arithmetic intensity I (flops/byte) is determined by the workload characteristics. Workloads between these two extremes are classified as “mixed”.

The total power of the system is modeled by Equation 7. The constant power C_0 comes from the uncore and other components in other VF domains. The static component C_1V models the leakage power, and the dynamic component $CV^2f\alpha$ (represented by C_2V^3) models the dynamic power in the cores.

$$Power = (C_0 + C_1V + C_2V^3) \times \frac{1}{\eta}. \quad (7)$$

The system energy efficiency is calculated by $\frac{Performance}{Power}$. Figure 2 shows the instantaneous system energy efficiency *v.s.* the voltage (no DVFS transitions). For compute bound workloads under a constant regulator efficiency, there exists a maximum energy efficiency within the typical voltage range (*e.g.*, $0.6V$ to $1.1V$), while for memory bound workloads, the maximum energy efficiency is achieved at the lowest voltage. A system with on-chip switching regulators usually has a regulator efficiency below 80% (green-line). For an on-chip LDO based system, however, the regulator efficiency $\eta = \frac{V}{V_{in}}$ varies with V and V_{in}. If input voltage V_{in} is fixed at 1.1V (the smooth line in red), the system is energy efficient only at high voltages close to V_{in}, and thus LDOs are undesirable for wide range DVFS adjustments. On the other hand, if multiple input voltages are allowed (the broken line in red), and LDOs are limited to switch within only 100mV below each V_{in}, then LDO efficiency can be guaranteed to fall within the 90% to 100% range. Multiple input voltages can come from the high efficiency off-chip switching regulators, which requires a framework with a hybrid hierarchy of regulators. The tradeoffs involved in designing such a framework are discussed in the next section.



(a) Compute-bound workloads (b) Memory-bound workloads

Figure 2. The system energy efficiency varies with voltage and regulator efficiency (eff). The smooth red-line shows the system is energy efficient only at high voltages if a fixed V_{in} of 1.1V is provided for the LDO. The broken red-line shows the system can maintain high energy efficiency over a wide voltage range when a series of input voltages are available for the LDO. The system energy efficiency with on-chip switching regulators falls below the green-line (eff=0.8).

3.2 Opportunities for Nanosecond DVFS

To save energy with DVFS, it is crucial to identify the execution intervals in which the performance requirements can be met while lowering the voltage and frequency levels. These intervals are defined as *Candidate Intervals*. Figure 3 shows the energy dissipation during representative execution intervals for workloads *art*, *fft*, and *swim* [1]. 1000 intervals are plotted, where each interval is measured by $1K$ committed instructions on a core. An interval with peak energy is typically a candidate interval, in which the core may either take much longer time to finish a given amount of work (*e.g.*, in memory intensive phases), or consumes high power but the performance does not improve commensurately (*e.g.*, the peak energy efficiency in compute-bound workloads may not appear at the peak voltage, Figure 2 (a)). On-chip regulators are able to switch in every interval (∼100ns), while off-chip regulators can only switch every 200 intervals (10-100μs), which causes many energy saving opportunities to be lost. The ability to identify sufficiently fine grained intervals with on-chip regulators therefore provides a greater opportunity to exploit energy-delay tradeoffs.

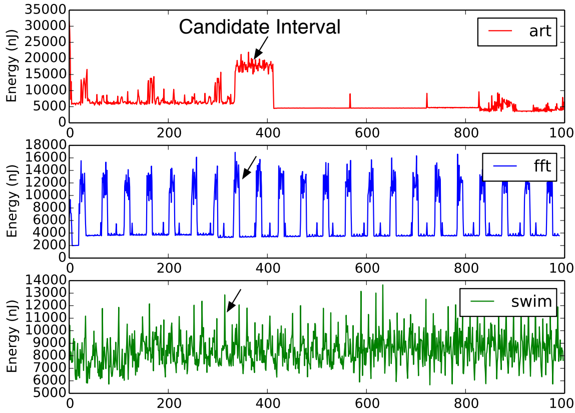


Figure 3. Energy in 1000 intervals ($1K$ instructions per interval). On-chip regulators allow fine-grained DVFS in every interval (∼100ns), while off-chip regulators support coarse-grained control every 200 intervals (10-100μs).

4. Voltage Conversion Framework

Conventional voltage conversion frameworks typically employ either a single off-chip switching regulator (high efficiency but coarse-granularity), or multiple on-chip switching regulators (fine granularity but low efficiency). To enable power control with both fine granularity and high regulator efficiency, this section explores a hierarchical M-N framework with two different types of regulators. Three variations of this framework are discussed and compared in terms of performance and cost.

4.1 M-N Schemes

Figure 5 shows M off-chip switching regulators that provide M distinct voltage levels and are connected to N per-core on-chip LDOs, each of which can adjust its output voltage within a limited range relative to its input voltage. Off-chip switching regulators achieve a high efficiency but can only respond in microseconds. The limited ratio of the output voltage to the input voltage ensures that the LDOs can also maintain a high energy conversion efficiency. A switching fabric—implemented with transistor switches, and capable of responding in nanoseconds with reasonable cost [53]—realizes the full connections between the off-chip and on-chip regulators. Different values of M and N result in different cost and performance tradeoffs, as will be discussed next.

1-N Scheme. This scheme uses a single (M=1) off-chip switching voltage regulator, no switching fabric, and N on-chip LDO regulators. Similarly to [45], in which the off-chip switching regulator is in charge of slow, wide range voltage adjustments; each LDO has the freedom to fast tune the voltage in each core within a small voltage range. Since all of the LDOs share the same input voltage, this framework is suitable only for well balanced parallel applications

[1] The experimental setup is discussed in Section 6

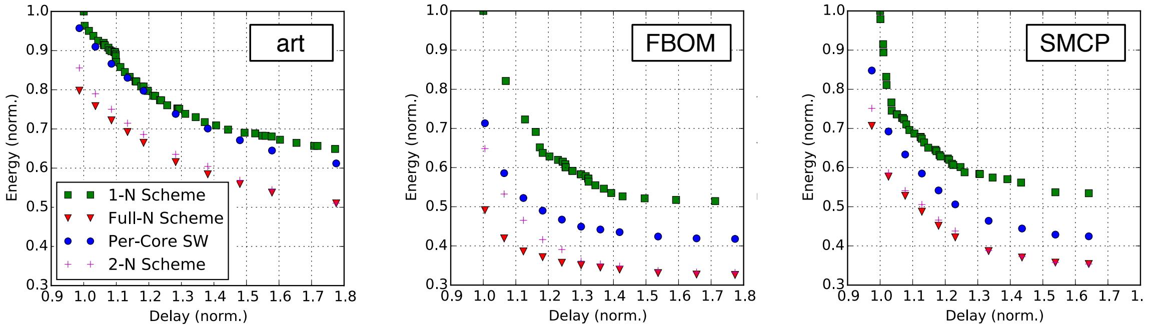


Figure 4. Energy-delay Pareto plots for four voltage conversion frameworks. ART is multithreaded, and FBOM (fft, bt, ocean, mg), and SMCP (swim, mg, cg, sp) are multiprogrammed workloads. The energy and delay are normalized to 1-N scheme. Full-N constitutes the optimal boundary when both fine-granularity and near-perfect regulators are available. Baseline "Per-Core SW" is a framework with only on-chip switching regulators per core (η up to 80%).

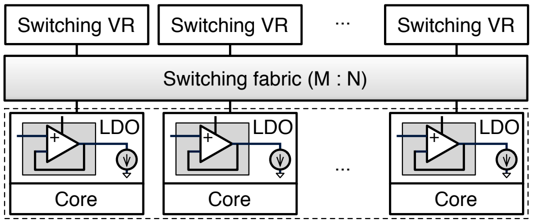


Figure 5. A hybrid voltage conversion framework, where M off-chip switching regulators (η=1, microseconds response) and N on-chip LDOs (η=0.9∼1 if V_{out}=0.9V_{in}∼V_{in}, nanoseconds response) are connected via a switching fabric.

in which core-to-core variations in voltage demand are relatively small [45]. It fails to handle cases efficiently in which different cores exhibit dissimilar behaviors.

Full-N Scheme. This scheme employs M off-chip switching regulators, and N on-chip LDO regulators. M should be large enough such that each switching regulator provides a distinct voltage, and together the switching and LDO regulators can cover the entire voltage range. Each on-chip LDO is free to choose one voltage as its input through the switching fabric. Table 2 shows M=3 is sufficient to cover the full voltage range (*e.g.*, typically 1.1V to 0.72V at 22nm). Therefore, a lower bound in regulator efficiency of 90% can be guaranteed. This scheme achieves both fine granularity and high regulator efficiency, but incurs a high area cost due to too many off-chip switching regulators.

Input voltage	Step size	Output range	Efficiency
1.1V	50mV	1.1V to 1.0V	min. 90%
0.95V	45mV	0.95V to 0.86V	min. 90%
0.8V	40mV	0.8V to 0.72V	min. 90%

Table 2. A list of voltage ranges for LDOs.

2-N Scheme. This scheme is a trade-off between the two previous extremes. Two (M=2) off-chip switching regulators provide two distinct voltages that partially cover the full voltage range. N per-core LDOs can freely choose between these voltages as inputs. However, whenever one or more cores request a voltage out of the currently covered range, the result is a "voltage miss", and one of the off-chip switching regulators may need to change its voltage, which takes microseconds. A simple "replacement" policy—*majority vote*—that greatly reduces the performance penalty is employed in this work. The off-chip switching regulators always provide two voltages that receive the most votes. The unserved cores are assigned a currently available voltage that is higher and closest to their requested voltages. The 2-N scheme opportunistically achieves fine-grained power control with a high regulator efficiency.

4.2 Energy-Delay Pareto Frontier

The design space exploration (Figure 4) for three frameworks, as well as a baseline framework with only on-chip switching regulators (shown as "Per-Core SW" with a practical efficiency up to 80%, *a la* the Intel Haswell processors [18]) are explored. A step-wise $0/1$ integer linear programming approach is used to generate the energy-delay points for each program.The energy-delay Pareto curves for the Full-N scheme constitute the lower bounds. These Pareto curves are close to the ideal case with fine-grained power control and perfect regulators, but the high cost of multiple off-chip regulators may render the Full-N scheme impractical. The cost-effective scheme 2-N achieves results close to Full-N, partially due to the infrequent voltage miss, and the inexpensive energy penalty (since unsatisfied cores will be assigned an available voltage closest to their optimal voltage). *Per-Core SW* performs better than 1-N for multiprogrammed workloads due to the fine-grained power control at each core, but performs worse than 2-N and Full-N, due to its lower regulator efficiency.

5. RL-based Control Policy

DVFS control policies can be decoupled from the proposed framework. *Ad hoc* policies that are unaware of regulator efficiencies, such as simple feedback-based policies or sophisticated, statistical policies are evaluated in Section 7. This section presents a new, scalable control policy that is aware of the regulator efficiency characteristics to better manage the system power.

5.1 Top Level Power Budget Allocation

To achieve a scalable control policy, this paper adopts a power bidding approach [54] for top-level power budget allocation. The power budget P_{total} is partitioned among N cores, and the power P_i assigned to the ith core is determined by:

$$P_i = \frac{b_i}{\sum_{i=1}^{N} b_i} \times P_{total} \tag{8}$$

where b_i is the bid raised from core i (reflecting the performance utility of that core). The bid is determined by the the fairness and the instruction throughput losses compared to the last measurement. Therefore, the bid for the ith core is defined as:

$$b_i = \frac{ips\_loss_i}{fairness_i} \tag{9}$$

where ips is instructions per second, and the $fairness_i$ is defined as the local relative instruction throughput:

$$fairness_i = \frac{ips_i}{\sum_{i=1}^{N} ips_i} \tag{10}$$

As a result of this construction, a slow agent that experiences a performance degradation will raise its bid; otherwise, the agent will keep a minimum bid.

5.2 Per-Core RL Agent

As introduced in Section 2 and depicted in Figure 6 (b), the power manager of the system is modeled by the RL agent. It uses system statistics as states and energy consumption as the reward ❶, and makes decisions about which voltage and frequency action to take ❷. To learn from experience, the RL agent internally maintains a *State-Action Mapping Table* that follows the Q-leaning update rule ❸. The entire process happens at each sampling interval (step), which is $1K$ instructions committed by the local core. The large number of sampling intervals obtained by the nanosecond sampling enables the RL agents to quickly converge to high quality control policies. RL agents are independent in making decisions, but also compete for the total power budget and cooperate by occasionally sharing experiences to speedup the learning process.

5.2.1 Problem Formulation

Actions. A voltage and frequency pair is defined as an action. The voltage and frequency space in this paper is divided into three ranges and nine levels, with a step size of 50-40 mV (Table 2). The frequency varies from 4 GHz to 800 MHz.

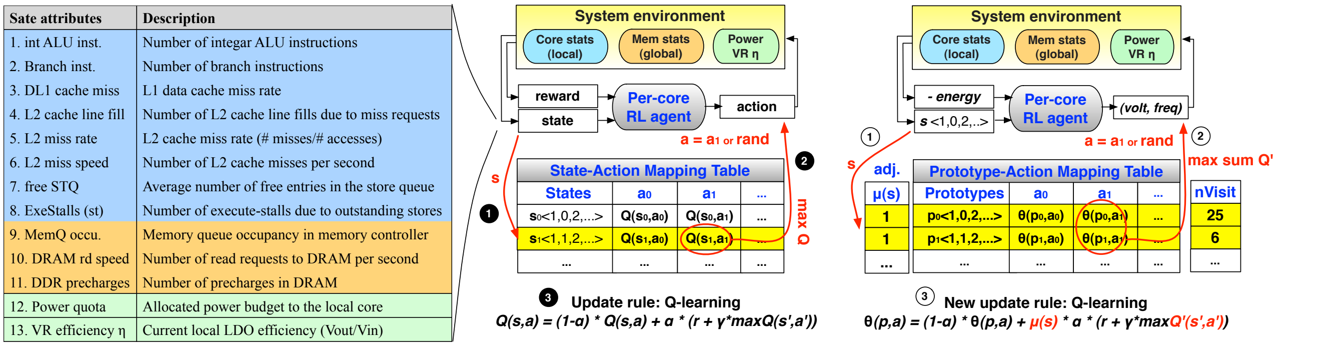
State Space. A vector of attributes is defined as a system state s, as shown in Figure 6 (a). The last two attributes (12 and 13) are the allocated power budget and the VR efficiency, while all of the other attributes are collected from performance counters. Attributes 1-2 represent runtime instruction composition, and are used to categorize an interval as *compute-intensive*, *compute-branch*, or *memory-intensive*. Attributes 3-6 describe cache behavior, and 7-8 are related to store instructions. Attributes 9-11 are statistics from the memory controller, and provide a closer look at memory events; large values indicate that the interval may be a candidate interval to lower voltage and frequency levels. To distinguish the memory activity incurred by local events and the possible contention from other cores, attributes 4-6 are reserved for local memory statistics, while 9-11 are global. As a result, RL agents are aware of contention, and can take actions accordingly. Resource allocation and contention resolution are separate issues and have been addressed in recent works [5, 54].

Reward. The immediate reward r is assigned a value of $-1\times$*energy*. Reinforcement learning is formulated to maximize the cumulative reward (total negative energy). As a result, maximizing total reward is equivalent to minimizing the total energy. The energy in an interval can be computed by *power* × *time*. The power can be measured or estimated using existing power models. The performance loss is computed between adjacent intervals. In cases when undesired performance losses are observed, the reward is assigned a large negative number (*i.e*, penalty) to prevent the losses. The initial rewards start at 0.

State-Action Mapping Table. This table records past experiences for all possible states and actions, and is the core of the online Q-learning process. An example is shown in Figure 6 (b). State s is used to query the table at the RL-agent ❶; the action a_1 has the maximum Q-value (highest utility) in the matched entry, and thus is selected ❷. In addition, a random action may be selected with a small probability of ϵ to ensure continual design space exploration (the ϵ-greedy policy is described in Section 2.3). After taking the action, the system transits to state s'. By following the Q-learning update rule ❸, a Q-value is learned from the past experience (the old Q-value), and the total reward that incorporates the immediate reward r that results from the current action a and the discounted future reward $\gamma \cdot max_{a'} Q(s', a')$.

Learning Parameters. Parameters are tuned offline iteratively and empirically. For example, ϵ in the ϵ-greedy policy is used for balancing exploration and exploitation. In this work, an agent with $\epsilon > 0.1$ explores more, and finds out optimal actions earlier, but performs worse in the long run because it often gets stuck in suboptimal actions. An agent with $\epsilon < 0.01$ improves slowly, but eventually performs better. The final selection of $\epsilon = 0.05$ is a trade-off between speed and long term results. Section 7 provides more detailed discussions on tuning the learning rate α for speed, and discount rate γ for learning preference.

Energy-Delay Space Exploration. By adjusting the aforementioned penalty values, the performance constraint can be altered. A small penalty guides the RL agent towards the lowest energy with unconstrained performance loss. A large penalty drives the policy towards the lowest energy without a significant performance loss. Any penalty value in between will guide the online learning to find the low-



Sate attributes	Description
1. int ALU inst.	Number of integar ALU instructions
2. Branch inst.	Number of branch instructions
3. DL1 cache miss	L1 data cache miss rate
4. L2 cache line fill	Number of L2 cache line fills due to miss requests
5. L2 miss rate	L2 cache miss rate (# misses/# accesses)
6. L2 miss speed	Number of L2 cache misses per second
7. free STQ	Average number of free entries in the store queue
8. ExeStalls (st)	Number of execute-stalls due to outstanding stores
9. MemQ occu.	Memory queue occupancy in memory controller
10. DRAM rd speed	Number of read requests to DRAM per second
11. DDR precharges	Number of precharges in DRAM
12. Power quota	Allocated power budget to the local core
13. VR efficiency η	Current local LDO efficiency (Vout/Vin)

System environment
Core stats (local)
Mem stats (global)
Power VR η
reward
state
Per-core RL agent
action
a = a1 or rand
s
1
2
max Q
State-Action Mapping Table
States | a0 | a1 | ...
s0<1,0,2,...> | Q(s0,a0) | Q(s0,a1) | ...
s1<1,1,2,...> | Q(s1,a0) | Q(s1,a1) | ...
... | ... | ... | ...
3
Update rule: Q-learning
Q(s,a) = (1-α) * Q(s,a) + α * (r + γ*maxQ(s',a'))
System environment
Core stats (local)
Mem stats (global)
Power VR η
- energy
s <1,0,2,..>
Per-core RL agent
(volt, freq)
1
2
a = a1 or rand
s
max sum Q'
adj.
μ(s)
1
1
...
Prototype-Action Mapping Table
Prototypes | a0 | a1 | ...
p0<1,0,2,...> | θ(p0,a0) | θ(p0,a1) | ...
p1<1,1,2,...> | θ(p1,a0) | θ(p1,a1) | ...
... | ... | ... | ...
nVisit
25
6
...
3
New update rule: Q-learning
θ(p,a) = (1-α) * θ(p,a) + μ(s) * α * (r + γ*maxQ'(s',a'))

(a) Attributes used in system state space (b) Q-learning Procedure (c) Updated Q-learning with Kanerva Coding

Figure 6. The proposed reinforcement learning approach in power management.

est energy under a specific performance constraint. Figure 7 shows a static regression model derived from simulation. It depicts how the penalty is related to performance constraints, and can be used at runtime to select on appropriate penalty value.

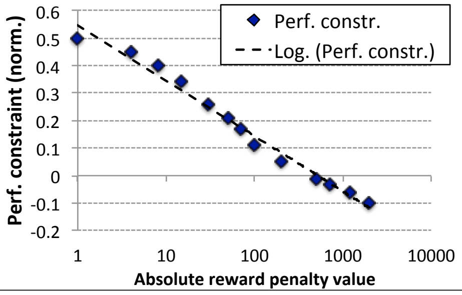


Figure 7. A regression model between penalty rewards to specified performance constraints.

Experience Sharing. RL agents learn from experience. An experience sharing approach—"Max share" [30]—is employed to speedup the learning process. A shared copy of the Prototype-Action Mapping table is maintained in memory. Each core takes turns to check the shared copy every $10K$ intervals. Since each prototype maintains a $nVisit$ counter that tracks its learning history, the θ-values of a more experienced prototype with a larger $nVisit$ will get copied from a local table to the shared table, or vice versa.

5.2.2 Feature Selection

Feature selection is employed at design time to select a subset of the system attributes when constructing the system state. To arrive at the state attributes in Figure 6 (a), an initial attribute space with 2500 features is searched by a best-first algorithm [58], and the utility of each subset is evaluated by a *Cfs Subset Evaluator* [17]. The most useful subset is chosen as the state representative used by the reinforcement learning. By reducing the dimensionality of the initial state space and removing redundant data points, it also limits over fitting, and facilitates the training process.

5.2.3 Adaptive Kanerva Coding

As discussed above, the State-Action Mapping Table records the Q-values of all possible states and actions. The table size can be exponentially large when each attribute has multiple values, and this poses a significant challenge to reinforcement learning algorithms. Kanerva Coding [1] is an approximation approach to reduce the complexity of high dimensionality, and is used in this work. As shown in Figure 6 (c), the State-Action Mapping Table is replaced with a *Prototype-Action Mapping Table* that incorporates only a subset of the states.

Prototype. A state that is selected to be the representative of a subset of states is called a *prototype* p. A state s is *adjacent* to a prototype p, if they are the same, or differ by only one element. For example, state <1,0,2> is adjacent to prototype <1,1,2>, but not to prototype <1,2,3>.

Prototype-Action Mapping Table. A set of chosen prototypes constitute the Prototype-Action Mapping Table (Figure 6 (c)), which is substantially smaller than the State-Action Mapping Table that includes all possible states and actions. Similarly to conventional Q-learning, the table entries record past experiences using θ-values to approximate Q-values. State s is used to query the table ①, and every adjacent prototype is considered a match. The adjacency, $\mu_i(s)$, is 1 if the state s is adjacent to a prototype p, and is 0 otherwise. The action a_1 that has the maximum approximate Q'-value (given by the sum of all adjacent θ-values) is chosen ②. The Q-learning update rule is modified to incorporate Kanerva Coding ③.

Adaptive Kanerva Coding. The size of the table and the distribution of the chosen prototypes in the state space affect the performance of Kanerva coding.

Two principles are respected using an adaptive approach [1] to guarantee high performance: (1) no prototype is visited too often to prevent *collisions*—cases where too many states are adjacent to one prototype, so the prototype cannot distinguish among different states; and (2) no prototype is visited too infrequently.

- *Replacement.* Each prototype maintains a counter $nVisit$, to count its visit frequency. When a state query results in a miss in the table, *i.e.*, no adjacent prototype is found, the prototype with the smallest $nVisit$ is replaced.
- *Deletion and split.* To avoid being stuck with a few "hot" prototypes, and to remove rarely visited prototypes, these

prototypes are periodically deleted or split, as enforced by a *update rate*. After deleting a "cold" entry, the most visited prototypes are split into two adjacent prototypes by randomly incrementing one of the attributes.

5.2.4 Hardware Implementation

Learning and making predictions at a $1K$-instruction interval imposes a latency requirement on the algorithm. To address this need, a hardware control unit is implemented and fully pipelined at the processor core clock frequency. Similarly to prior work [20], the control logic is divided into a 5-stage pipeline, where *stage-1* is used to sense and discretize attributes from the system; *stage-2* is used to query θ-values and compute adjacencies; *stage-3* is used to calculate the approximate maximum Q' value; and *stage-4* and *stage-5* are used to update θ-values in the Prototype-Action Mapping Table. Different from prior work [20], the approximation method—adaptive Kanerva coding—requires a much smaller table for the prototypes. A local table of 64 entries, which takes less than 1KB SRAM storage, is sufficient. The pipeline logic is implemented in Verilog RTL and fully synthesized at 45nm; the results are scaled to 22nm using the methodology described in prior work [14]. The area for an RL agent (including statistics preprocessing logic) is $284\mu m^2$, and the peak power is $0.89mW$, which makes it feasible for the RL agent to be implemented in a local core.

6. Experimental Setup

We evaluate the system energy efficiency ($\frac{1}{energy}$) under a 5% performance loss constraint (relative to running at the highest voltage and frequency). We present two levels of comparisons: 1) comparison among voltage conversion frameworks (Table 3) with the same oracle control policy (the baseline framework is "PerCore SW" using only on-chip switching regulators); and 2) comparison among different control policies on the same framework (2-N). The baseline policy is a coarse-grained optimal control policy ("Coarse-optimal"). Simple *ad hoc* policies that are unaware of regulator efficiency characteristics are also evaluated.

Frameworks	On-chip VR	Off-chip	Granularity	Cost
1-N	LDO η >0.9	1 SW	coarse	low
2-N	LDO η >0.9	2 SWs	limited fine	med.
Full-N	LDO η >0.9	3 SWs	fine	high
PerCore SW	SW η <0.8	None	fine	med.

Table 3. Voltage conversion framework configurations. On-chip voltage regulators are per core, and the baseline "PerCore SW" uses only on-chip switching regulators (SW).

On-demand policy [38]. This is a simple feedback based control policy that is similar to the commercially used *CPUfreq* policy in Linux governors, and is ported to operate at the nanosecond granularity. This policy is based on the per-core utilization: when the utilization exceeds an *up threshold*, the core frequency is raised to the maximum; if the utilization drops below a *down threshold*, the core frequency is decreased by a predefined amount (*e.g.*, 20%).

PPEP-like policy [50]. This is a recent, sophisticated, modeling based control policy. Essentially it includes four analytical and statistical models: a performance model, an idle power model, a dynamic power model, and a power prediction model that predicts power for voltage-frequency levels different from the current level. A one-step capping strategy is used to select a voltage-frequency level that maximizes the performance under a power cap or minimizes the power under a performance constraint.

Proposed RL policy. The proposed reinforcement learning policy (Section 5) is tuned via a series of sensitivity studies (Section 7), and initialized as follows: all θ-values in the Prototype-Action Mapping Table are set to 0; the learning rate α is set to 0.1; the discount rate γ is set to 0.95; the probability ϵ in the ϵ-greedy policy is set to 0.05; and the system is initialized to the highest voltage and frequency.

6.1 Architecture

A four-core OoO processor with one hardware context per core, and two DDR3 memory channels is simulated with a heavily modified SESC [42] simulator. Core parameters are presented in Table 4, and the voltage-frequency pairs are shown in Table 2. The processor power is evaluated with McPAT [33] at 22nm, and main memory power is derived from a Micron data sheet [35] for DDR3-1066. The RL agents are implemented in RTL with Cadence NCSim [6] and synthesized with the Synopsys Design Compiler [10] at 45nm using the NanGate standard cell library [36]. The numbers are scaled to 22nm using the scaling factors reported in [14]. The 0/1 integer linear programming in the design space exploration is evaluated with a linear solver, Gurobi [16]. The feature selection for the reinforcement learning is performed with WEKA [56].

Architecture parameters	Values
Base voltage-frequency	1.1 V-4 GHz
Fetch, rename, dispatch width	4, 4, 4
Issue, commit, writeback width	6, 4, 4
Branch prediction	combined branch predictor
BTB, RAS	1024 4-way, 1024
Fetch, issue, load, store queue	48, 64, 48, 32
ALU, FPU, load, store unit	6, 6, 4, 4
Reorder buffer	128
Physical register files	int256, fp256
iTLB, dTLB	32-entry, 4-way, 32KB
Private L1 instruction cache	32KB, 8-way, 64B
Private L1 data cache	32KB, 8-way, 64B
Shared L2 cache	8MB, 8-way, 64B
Hardware prefetcher	stream prefetcher
Coherence protocol	MESI
Main memory	FR-FCFS, open-page policy

Table 4. Architectural simulation setup

6.2 Applications

A mix of 14 parallel and 13 multiprogrammed workloads are evaluated (Table 5 and 6). Parallel benchmarks are from the NAS OpenMP [2], NuMineBench [40], Phoenix (pthread) [60] , SPEC OpenMP [11], and SPLASH-2 [57] suites. Single threaded benchmarks are from the SPEC

2006 [19] and Phoenix (sequential) [60] suites, and are combined to form multiprogrammed workloads.

Parall. Benchmarks	Suite	Input
SCALPARC	NuMineBench	F26-A32-D125K.tab
CG	NAS OpenMP	Class A
MG	NAS OpenMP	Class A
SP	NAS OpenMP	Class A
BT	NAS OpenMP	Class A
ART-OMP	SPEC OpenMP	MinneSpec-Large
SWIM-OMP	SPEC OpenMP	MinneSpec-Large
MATRIX MULT.	Phoenix	3000 x 3000 matrix
HISTO	Phoenix	small
FFT	SPLASH-2	2^{20} complex data
CHOLESKY	SPLASH-2	tk29.O
OCEAN	SPLASH-2	514×514 ocean
RAYTRACE	SPLASH-2	car.env
FMM	SPLASH-2	input.2048

Table 5. Parallel applications.

MP. Benchmarks	Composition & Inputs
401.BZIP2	4×BZIP2, chicken.jpg (SPEC06)
403.GCC	4×GCC, 166.i (SPEC06)
429.MCF	4×MCF, inp.in in train (SPEC06)
470.LBM	4×LBM, 100-100-130-ldc.of (SPEC06)
433.MILC	4×MILC, su3imp.in (SPEC06)
462.LIBQUANTUM	4×LIBQUANTUM, 143 25 (SPEC06)
SPEC-BBML	2×BZIP2, MCF, LBM
SPEC-QBBM	LIBQUANTUM, 2×BZIP2, LBM
SPEC-QQBM	2×LIBQUANTUM, BZIP2, MCF
PHOENIX-HLWP	HISTO, Linear, WordCount, PCA
PHOENIX-HWSP	HISTO, WordCount, StringMatch, PCA
PHOENIX-HSPK	HISTO, StringMatch, PCA, KMEANS
PHOENIX-SSPP	2×StringMatch, 2×PCA

Table 6. Multiprogrammed applications.

7. Evaluation

This section presents energy efficiency for different frameworks and control policies. The parameter tuning process is presented with a series of sensitivity studies.

7.1 Energy Efficiency of DVFS Frameworks

Figure 8 depicts the energy efficiency results achieved by the four frameworks presented in Table 3, as well as an "Ideal" framework with perfect regulators and fine-granularity in space and time.

The 1-N scheme achieves a high regulator efficiency, but all of the cores are locked to the same small voltage range. As a result, 1-N performs well only on parallel workloads such as *matrix multiply*, *cholesky*, *sp*, and *fmm*, which are well balanced and exhibit little core-to-core variability in voltage.

The baseline "Typical Per-Core SW" scheme permits flexibly tuning the voltage and frequency in space and time, but suffers from a low regulator efficiency ($<$ 0.8). As a result, except for workloads that exhibit fine-grained phases with disparate voltage and frequency levels, such as *fft*, *scalparc*, *cg*, and *swim*, it performs worse than several of the other frameworks with greater regulator efficiencies.

The Full-N and "Ideal" frameworks enjoy the fine-granularity in space and time, maintain a high regulator efficiency (η=0.9$\sim$1 for Full-N, η=1 for "Ideal"), and perform better than the baseline. However, due to the high cost of multiple off-chip regulators, Full-N may not be practical.

The 2-N scheme achieves a high regulator efficiency, while allowing power control within the full voltage range with a limited flexibility. It performs close to Full-N in most cases where workload behavior is not overly diverse. In some cases, such as *cg*, the optimal voltage and frequency levels fluctuate among 3 ranges, which leads to frequent "voltage misses" and makes 2-N perform worse than Full-N. On average, 2-N delivers 18% higher energy efficiency as compared to the baseline.

7.2 Energy Efficiency of Control Policies

On the selected framework 2-N, four DVFS control policies are evaluated, as shown in Figure 9. The baseline "Coarse-Optimal" uses an oracle control policy at the microsecond granularity, and thus sets an upper bound on coarse-grained control. All other policies assume nanosecond fine-grained control.

The on-demand policy is a simple, feedback-based policy that prioritizes performance. It does not perform well for Phoenix benchmarks, such as *histogram, HWSP*, and *HSPK* mix, and memory bound benchmarks, such as *470.lbm* and *429.mcf*. The memory events reduce the activity, but do not necessarily decrease the core utilization. Therefore, the maximum frequency is selected to improve the marginal performance, but opportunities to save more energy are lost.

The PPEP-like policy is heavily dependent on the accuracy of the models. One of the key observations in PPEP [50] is that the lowest energy is achieved at the lowest voltage and frequency level, which is not true in our observations for compute bound workloads (Figure 2 in Section 3). This inaccuracy makes it miss energy saving opportunities for workloads involving abundant computations, such as *matrix-multiply*, and workloads with a fair mix of computation and memory activity, such as *fft, milc, scalparc*, and *swim*.

The RL policy relies on experience to learn an optimal policy on-the-fly; as a result, workloads exhibiting repeated patterns converge faster and achieve the most energy savings, such as *fft*, *cg*, and *swim*. These workloads also have fine-grained phases (Figure 3) that make the nanosecond RL policy effective in saving energy as compared to the coarse-grained baseline. Multiprogrammed workloads in which each core runs the same application (*403.gcc, 433.milc*) or applications with similar behavior (*PHOENIX-HSPK*) exhibit good results, because the learned policy is universal, and captures the common characteristics. However, workloads such as *429.mcf* and *470.lbm* save little energy over the baseline. They exhibit long memory intensive phases; as a result, the coarse grained policy is sufficiently effective. On average, the RL policy improves the energy efficiency by 21% over the baseline.

7.3 Online Energy-Delay Space Exploration

The performance constraint for the proposed RL policy is adjusted on demand. To minimize the energy under a specific delay, the reward for the RL agent must be set accord-

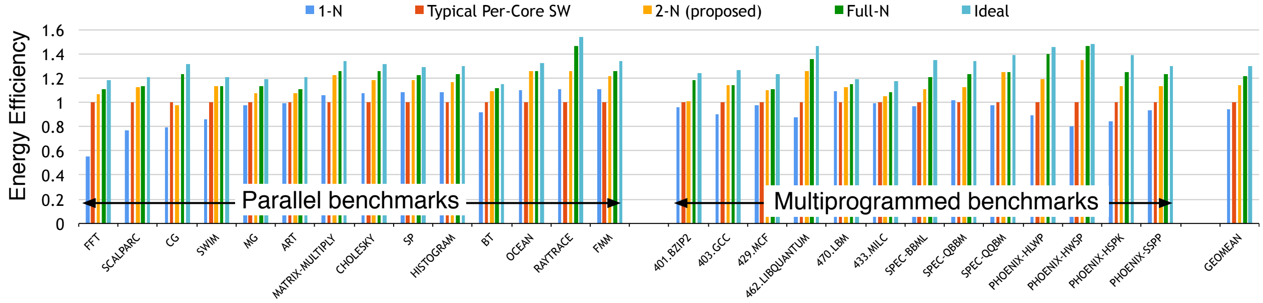


Figure 8. Energy efficiency (1/Energy) under a maximum of 5% performance loss is normalized to the baseline "Typical Per-Core SW"—a framework with only on-chip switching regulators ($\eta < 80\%$). The same oracle control policy is assumed for all frameworks.

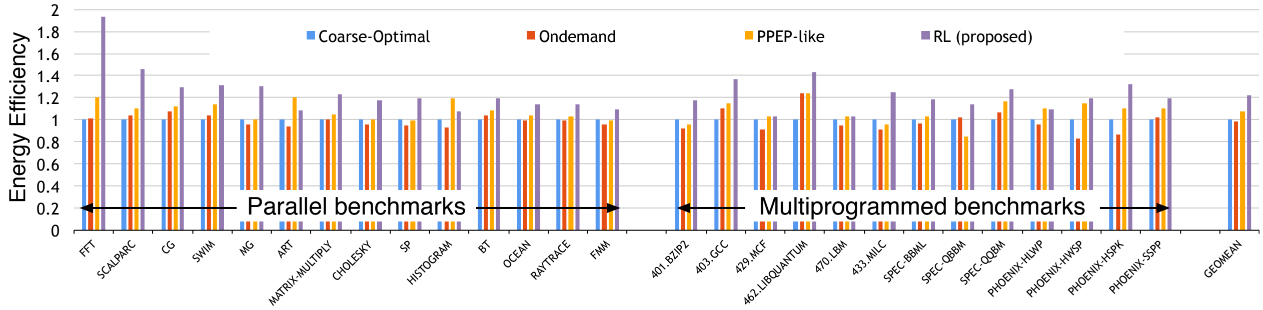


Figure 9. Energy efficiency (1/Energy) normalized to the baseline "Coarse-optimal"—a coarse-grained control policy with perfect knowledge of the workloads. All policies are running on the same framework 2-N under a maximum 5% performance loss constraint.

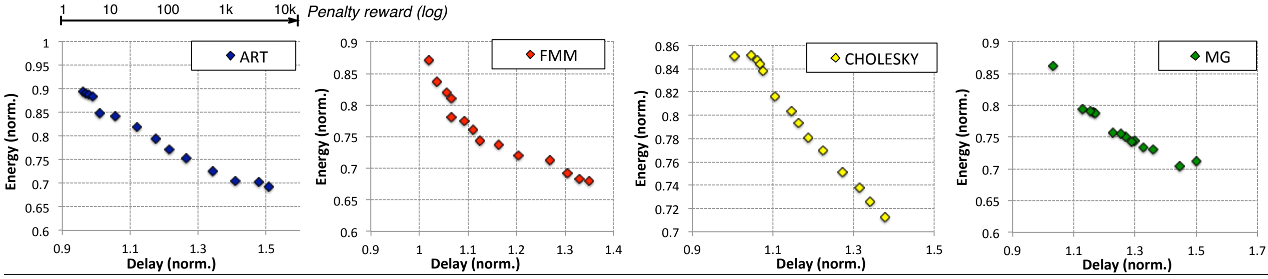


Figure 10. Online energy-delay space exploration by adjusting the reinforcement learning penalty reward from 1 to 10k.

ingly, and this is based upon a static regression model (Section 5). Figure 10 shows the energy-delay for four representative workloads. The relationship between the performance constraint and the log scale penalty reward is modeled by linear regression, suggesting that the energy consumption can be minimized in different situations adaptively.

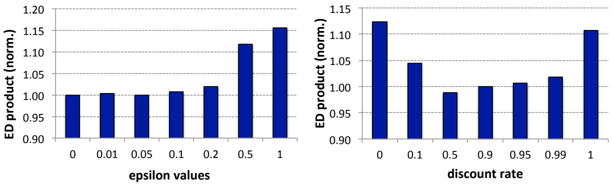


Figure 11. System ED product v.s. probability ϵ, and discount rate γ. Results are normalized to the setting ($\epsilon = 0.05$, $\gamma = 0.9$) in this work.

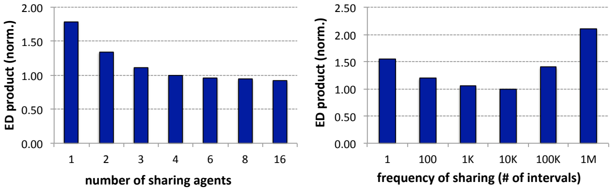


Figure 12. System ED product v.s. the number of RL sharing agents (cores), and the sharing frequency. Results are normalized to the setting (4 agents, $10K$ intervals) in this work.

7.4 Sensitivity Study for RL Control Policy

The parameter tuning process for reinforcement learning and Kanerva coding is explained in this section.

7.4.1 Sensitivity to RL parameters

ϵ-Greedy Policy. The plot on the left hand side of Figure 11 shows the relationship between the achieved system energy-

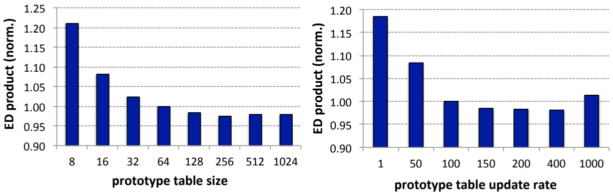


Figure 13. ED product v.s. the table size, and update rate. Results are normalized to the setting (64 entries, 100 intervals).

delay (ED) product and the ϵ values used in the ϵ-greedy policy. As ϵ increases, the *exploration vs. exploitation* balance is tilted toward exploration; as a result, a greater number of sub-optimal actions have to be taken along the learning process, resulting in a degradation in the ED product.

Discount Rate γ. The plot on the right hand side of Figure 11 shows the impact of varying the discount rate γ from 0 to 1, which places progressively greater emphasis on future rewards. In the extreme cases, where γ is 0 or 1, the learning mechanism becomes either too "myopic" (failing to establish connections to the ultimate goal) or too "farsighted" (failing to converge), which negatively affects the system ED product.

Experience Sharing. The plot on the left of Figure 12 shows the influence of varying number of RL agents. As expected, experience sharing speeds up the learning as the number of agents increases, since the agents start to apply the optimal policy earlier. However, a diminishing reward is observed for the system ED product, because the improvement stops bringing extra benefits after convergence. The plot on the right hand side of Figure 12 shows the impact of varying the sharing frequency. Too frequent sharing incurs increased memory traffic and communication that consumes extra energy. Too infrequent sharing is also less effective, since each agent becomes largely independent and takes less advantage of other's experiences.

7.4.2 Sensitivity to Kanerva Coding Parameters

Prototype-Action Mapping Table Size. The plot on the left of Figure 13 shows that an excessively small table limits the coverage of the state space, resulting in either too many "collisions" or frequent replacements, and ultimately a degradation of the system ED product. Increasing the size of the table typically improves performance; however, too many entries in the table increases the implementation overhead.

Table Update Rate. The plot on the right hand side of Figure 13 shows the impact of the update rate. An excessively high update rate causes thrashing at the table. Newly inserted prototypes may be deleted before they are warmed up. In contrast, an excessively low update rate causes the table to follow only the least-frequently-used rule, which works well in the early stages of design space exploration, but can be problematic in the later stages, where no unknown state is encountered and no new states are inserted.

8. Conclusions

On-die integration of voltage regulator modules enables precise and fine-grained DVFS power management, and holds significant potential to improve the energy-efficiency of future microprocessors. We observe that on-chip voltage regulator efficiency affects the achievable energy savings. Power management frameworks with on-chip regulators should therefore be aware of the regulator efficiency loss. We propose a voltage conversion framework that leverages both off-chip switching regulators and on-chip LDOs to enable fine-grained DVFS as well as a high regulator efficiency. On top of this framework, a reinforcement learning based control policy is proposed. It does not rely on accurate offline models, and instead learns and evolves an optimal policy at runtime. As compared to a conventional framework with only on-chip switching regulators, the proposed framework 2-N saves 18% in energy, and the proposed RL policy achieves 21% greater energy efficiency than an oracle coarse-grained control policy, both under a maximum of 5% performance loss. We conclude that the proposed framework and policy hold the potential to significantly improve the energy efficiency of future computer systems.

Acknowledgments

The authors would like to thank anonymous reviewers for useful feedback. This work was supported in part by NSF grant CCF-1533762.

References

[1] Martin Allen and Phil Fritzsche. Reinforcement learning with adaptive kanerva coding for xpilot game ai. In *Evolutionary Computation (CEC), 2011 IEEE Congress on*, pages 1521–1528. IEEE, 2011.

[2] David H Bailey, Eric Barszcz, John T Barton, David S Browning, Russell L Carter, Leonardo Dagum, Rod A Fatoohi, Paul O Frederickson, Thomas A Lasinski, and Rob S Schreiber. NAS parallel benchmarks. Technical report, NASA Ames Research Center, March 1994. Tech. Rep. RNR-94-007.

[3] Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 1998.

[4] Hee Rak Beom and Kyung Suck Cho. A sensor-based navigation for a mobile robot using fuzzy logic and reinforcement learning. *Systems, Man and Cybernetics, IEEE Transactions on*, 25(3):464–477, 1995.

[5] R. Bitirgen, E. Ipek, and J. F. Martinez. Coordinated management of multiple interacting resources in chip multiprocessors: A machine learning approach. In *International Symposium on Microarchitecture*, Lake Como, Italy, Nov 2008.

[6] Incisive enterprise simulator. `http://www.cadence.com/products/fv/enterprise_simulator`.

[7] Yu-Han Chang, Tracey Ho, and Leslie Pack Kaelbling. Mobilized ad-hoc networks: A reinforcement learning approach.

In *Autonomic Computing, 2004. Proceedings. International Conference on*, pages 240–247. IEEE, 2004.

[8] C.K. Chava and J. Silva-Martinez. A robust frequency compensation scheme for ldo regulators. In *Circuits and Systems, 2002. ISCAS 2002. IEEE International Symposium on*, volume 5, pages V–825–V–828 vol.5, 2002.

[9] Chia-Min Chen and Chung-Chih Hung. A capacitor-free cmos low-dropout voltage regulator. In *Circuits and Systems, 2009. ISCAS 2009. IEEE International Symposium on*, pages 2525–2528. IEEE, 2009.

[10] Design Compiler. Synopsys inc, 2000.

[11] L. Dagum and R. Menon. OpenMP: An industry-standard API for shared-memory programming. *IEEE Computational Science and Engineering*, 5:46–55, 1998.

[12] Qingyuan Deng, David Meisner, Abhishek Bhattacharjee, Thomas F Wenisch, and Ricardo Bianchini. Coscale: Coordinating cpu and memory system dvfs in server systems. In *Microarchitecture (MICRO), 2012 45th Annual IEEE/ACM International Symposium on*, pages 143–154. IEEE, 2012.

[13] Marco Dorigo and LM Gambardella. Ant-q: A reinforcement learning approach to the traveling salesman problem. In *Proceedings of ML-95, Twelfth Intern. Conf. on Machine Learning*, pages 252–260, 2014.

[14] Hadi Esmaeilzadeh, Emily Blem, Renee St Amant, Karthikeyan Sankaralingam, and Doug Burger. Dark silicon and the end of multicore scaling. In *Computer Architecture (ISCA), 2011 38th Annual International Symposium on*, pages 365–376. IEEE, 2011.

[15] Waclaw Godycki, Christopher Torng, Ivan Bukreyev, Alyssa Apsel, and Christopher Batten. Enabling realistic fine-grain voltage scaling with reconfigurable power distribution networks. In *Microarchitecture (MICRO), 2014 47th Annual IEEE/ACM International Symposium on*, pages 381–393. IEEE, 2014.

[16] Inc. Gurobi Optimization. Gurobi optimizer reference manual, 2015.

[17] Mark A Hall. *Correlation-based feature selection for machine learning*. PhD thesis, The University of Waikato, 1999.

[18] Per Hammarlund, Rajesh Kumar, Randy B Osborne, Ravi Rajwar, Ronak Singhal, Reynold D'Sa, Robert Chappell, Shiv Kaushik, Srinivas Chennupaty, and Stephan Jourdan. Haswell: The fourth-generation intel core processor. *IEEE Micro*, (2):6–20, 2014.

[19] John L. Henning. SPEC CPU2006 benchmark descriptions. *SIGARCH Comput. Archit. News*, 34(4):1–17, September 2006.

[20] E. Ipek, O. Mutlu, J. Martinez, and R. Caruana. Self-optimizing memory controllers : A reinforcement learning approach. In *International Symposium on Computer Architecture*, Beijing, China, Jun 2008.

[21] Engin Ipek, Onur Mutlu, José F Martínez, and Rich Caruana. Self-optimizing memory controllers: A reinforcement learning approach. In *Computer Architecture, 2008. ISCA'08. 35th International Symposium on*, pages 39–50. IEEE, 2008.

[22] Canturk Isci and Margaret Martonosi. Runtime power monitoring in high-end processors: Methodology and empirical data. In *Proceedings of the 36th annual IEEE/ACM International Symposium on Microarchitecture*, page 93. IEEE Computer Society, 2003.

[23] Rinkle Jain, Bibiche M Geuskens, Stephen T Kim, Muhammad M Khellah, Jaydeep Kulkarni, James W Tschanz, and Vivek De. A 0.45-1 v fully-integrated distributed switched capacitor dc-dc converter with high density mim capacitor in 22 nm tri-gate cmos. *IEEE Journal of Solid-State Circuits*, 49(4):917–927, 2014.

[24] Wonyoung Kim, D. Brooks, and Gu-Yeon Wei. A fully-integrated 3-level dc-dc converter for nanosecond-scale dvfs. *Solid-State Circuits, IEEE Journal of*, 47(1):206–219, Jan 2012.

[25] Wonyoung Kim, Meeta Sharma Gupta, Gu-Yeon Wei, and David Brooks. System level analysis of fast, per-core dvfs using on-chip switching regulators. In *High Performance Computer Architecture, 2008. HPCA 2008. IEEE 14th International Symposium on*, pages 123–134. IEEE, 2008.

[26] Jens Kober and Jan Peters. Reinforcement learning in robotics: A survey. In *Reinforcement Learning*, pages 579–610. Springer, 2012.

[27] S. Kose, E.G. Friedman, S. Tarn, S. Pinzon, and B. McDermott. An area efficient on-chip hybrid voltage regulator. In *Quality Electronic Design (ISQED), 2012 13th International Symposium on*, pages 398–403, March 2012.

[28] S. Kose, S. Tam, S. Pinzon, B. McDermott, and E.G. Friedman. Active filter-based hybrid on-chip dc-dc converter for point-of-load voltage regulation. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 21(4):680–691, April 2013.

[29] Selcuk Kose and Eby G. Friedman. On-chip point-of-load voltage regulator for distributed power supplies. In *Proceedings of the 20th Symposium on Great Lakes Symposium on VLSI*, GLSVLSI '10, pages 377–380, New York, NY, USA, 2010. ACM.

[30] R Matthew Kretchmar. Reinforcement learning algorithms for homogenous multi-agent systems. In *Workshop on Agent and Swarm Programming*, 2003.

[31] Hanh-Phuc Le, J. Crossley, S.R. Sanders, and E. Alon. A sub-ns response fully integrated battery-connected switched-capacitor voltage regulator delivering 0.19w/mm2 at 73% efficiency. In *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2013 IEEE International*, pages 372–373, Feb 2013.

[32] Ka Nang Leung and Philip KT Mok. A capacitor-free cmos low-dropout regulator with damping-factor-control frequency compensation. *Solid-State Circuits, IEEE Journal of*, 38(10):1691–1702, 2003.

[33] Sheng Li, Jung Ho Ahn, Richard D. Strong, Jay B. Brockman, Dean M. Tullsen, and Norman P. Jouppi. McPAT: An integrated power, area, and timing modeling framework for multicore and manycore architectures. In *International Symposium on Computer Architecture*, 2009.

[34] Amy McGovern, Eliot Moss, and Andrew G Barto. Scheduling straight-line code using reinforcement learning and rollouts. 1999.

[35] Micron Technology, Inc., http://www.micron.com//getdocument/?documentId=416. *8Gb DDR3 SDRAM*, 2009.

[36] NanGate FreePDK45 Open Cell Library. `http://www.nangate.com`.

[37] Junhong Nie and Simon Haykin. A Q-learning-based dynamic channel assignment technique for mobile communication systems. *Vehicular Technology, IEEE Transactions on*, 48(5):1676–1687, 1999.

[38] Venkatesh Pallipadi and Alexey Starikovskiy. The ondemand governor. In *Proceedings of the Linux Symposium*, volume 2, pages 215–230. sn, 2006.

[39] G. Patounakis, Y.W. Li, and Kenneth L. Shepard. A fully integrated on-chip dc-dc conversion and power management system. *Solid-State Circuits, IEEE Journal of*, 39(3):443–451, March 2004.

[40] J. Pisharath, Y. Liu, W. Liao, A. Choudhary, G. Memik, and J. Parhi. NU-MineBench 2.0. Technical report, Northwestern University, August 2005. Tech. Rep. CUCIS-2005-08-01.

[41] Krishna K. Rangan, Gu-Yeon Wei, and David Brooks. Thread motion: Fine-grained power management for multi-core systems. In *Proceedings of the 36th Annual International Symposium on Computer Architecture*, ISCA '09, pages 302–313, New York, NY, USA, 2009. ACM.

[42] Jose Renau, Basilio Fraguela, James Tuck, Wei Liu, Milos Prvulovic, Luis Ceze, Smruti Sarangi, Paul Sack, Karin Strauss, and Pablo Montesinos. SESC simulator, Jan 2005.

[43] Emre Salman and Eby G.Friedman. *High Performance Integrated Circuit Design*. McGraw-Hill Professional, 2012.

[44] Karan Singh, Major Bhadauria, and Sally A McKee. Real time power estimation and thread scheduling via performance counters. *ACM SIGARCH Computer Architecture News*, 37(2):46–55, 2009.

[45] A.A. Sinkar, H.R. Ghasemi, M.J. Schulte, U.R. Karpuzcu, and Nam Sung Kim. Low-cost per-core voltage domain support for power-constrained high-performance processors. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 22(4):747–758, April 2014.

[46] Abhishek A Sinkar, Hamid Reza Ghasemi, Michael J Schulte, Ulya R Karpuzcu, and Nam Sung Kim. Low-cost per-core voltage domain support for power-constrained high-performance processors. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 22(4):747–758, 2014.

[47] Vasileios Spiliopoulos, Stefanos Kaxiras, and Georgios Keramidas. Green governors: A framework for continuously adaptive dvfs. In *Green Computing Conference and Workshops (IGCC), 2011 International*, pages 1–8. IEEE, 2011.

[48] A Stafylopatis and K Blekas. Autonomous vehicle navigation using evolutionary reinforcement learning. *European Journal of Operational Research*, 108(2):306–318, 1998.

[49] N. Sturcken, E. O'Sullivan, Naigang Wang, P. Herget, B. Webb, L. Romankiw, M. Petracca, R. Davies, R. Fontana, G. Decad, I. Kymissis, A. Peterchev, L. Carloni, W. Gallagher, and K. Shepard. A 2.5d integrated voltage regulator using coupled-magnetic-core inductors on silicon interposer delivering 10.8a/mm2. In *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2012 IEEE International*, Feb 2012.

[50] Bo Su, Junli Gu, Li Shen, Wei Huang, Joseph L Greathouse, and Zhiying Wang. PPEP: Online performance, power, and energy prediction framework and dvfs space exploration. In *Microarchitecture (MICRO), 2014 47th Annual IEEE/ACM International Symposium on*, pages 445–457. IEEE, 2014.

[51] Gerald Tesauro. Online resource allocation using decompositional reinforcement learning. In *AAAI*, volume 5, pages 886–891, 2005.

[52] Chun-Yen Tseng, Li-Wen Wang, and Po-Chiun Huang. An integrated linear regulator with fast output voltage transition for dual-supply srams in dvfs systems. *Solid-State Circuits, IEEE Journal of*, 45(11):2239–2249, Nov 2010.

[53] Inna Vaisband, Burt Price, Seluk Kse, Yesh Kolla, EbyG. Friedman, and Jeff Fischer. Distributed ldo regulators in a 28 nm power delivery system. *Analog Integrated Circuits and Signal Processing*, 83(3):295–309, 2015.

[54] Xiaodong Wang and J.F. Martinez. Xchange: A market-based approach to scalable dynamic multi-resource allocation in multicore architectures. In *High Performance Computer Architecture (HPCA), 2015 IEEE 21st International Symposium on*, pages 113–125, Feb 2015.

[55] Samuel Williams, Andrew Waterman, and David Patterson. Roofline: An insightful visual performance model for multicore architectures. *Commun. ACM*, 52(4):65–76, April 2009.

[56] Ian H. Witten, Eibe Frank, Len Trigg, Mark Hall, Geoffrey Holmes, and Sally Jo Cunningham. Weka: Practical machine learning tools and techniques with java implementations, 1999.

[57] Steven Cameron Woo, Moriyoshi Ohara, Evan Torrie, Jaswinder Pal Singh, and Anoop Gupta. The SPLASH-2 programs: Characterization and methodological considerations. In *ISCA*, 1995.

[58] Lei Xu, Pingfan Yan, and Tong Chang. Best first strategy for feature selection. In *Pattern Recognition, 1988., 9th International Conference on*, 1988.

[59] Guihai Yan, Yingmin Li, Yinhe Han, Xiaowei Li, Minyi Guo, and Xiaoyao Liang. Agileregulator: A hybrid voltage regulator scheme redeeming dark silicon for power efficiency in a multicore architecture. In *High Performance Computer Architecture (HPCA), 2012 IEEE 18th International Symposium on*, pages 1–12. IEEE, 2012.

[60] Richard M Yoo, Anthony Romano, and Christos Kozyrakis. Phoenix rebirth: Scalable MapReduce on a large-scale shared-memory system. In *Proceedings of IEEE International Symposium on Workload Characterization*, 2009.