F-LEMMA: Fast Learning-based Energy Management for Multi-/Many-core Processors

An Zou*[1], Karthik Garimella*[1], Benjamin Lee[2], Christopher Gill[1], Xuan Zhang[1]
[1]{anzou, kvgarimella, cdgill, xuan.zhang}@wustl.edu; [2]leebcc@seas.upenn.edu
[1]Washington University in St. Louis; [2]University of Pennsylvania

ABSTRACT

Over the last two decades, as microprocessors have evolved to achieve higher computational performance, their power density also has increased at an accelerated rate. Improving energy efficiency and reducing power consumption is therefore of critical importance to modern computing systems. One effective technique to improve energy efficiency is dynamic voltage and frequency scaling (DVFS). In this paper, we propose F-LEMMA: a fast learning-based power management framework consisting of a global power allocator in userspace, a reinforcement learning-based power management scheme at the architecture level, and a swift controller at the digital circuit level. This hierarchical approach leverages computation at the system and architecture levels, and the short response times of the swift controllers, to achieve effective and rapid μs-level power management. Our experimental results demonstrate that F-LEMMA can achieve significant energy savings (35.2% on average) across a broad range of workload benchmarks. Compared with existing state-of-the-art DVFS-based power management strategies that can only operate at millisecond timescales, F-LEMMA is able to provide notable (up to 11%) Energy-Delay Product improvements when evaluated across benchmarks.

ACM Reference Format:
An Zou*[1], Karthik Garimella*[1], Benjamin Lee[2], Christopher Gill[1], Xuan Zhang[1]. 2020. F-LEMMA: Fast Learning-based Energy Management for Multi-/Many-core Processors. In *2020 ACM/IEEE Workshop on Machine Learning for CAD (MLCAD '20), November 16–20, 2020, Virtual Event, Iceland.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3380446.3430630

1 INTRODUCTION

Multi/manycore processors have become mainstream computing workhorses for both general-purpose and embedded systems. With the demise of Dennard scaling, high power density has become a key design constraint and a performance-limiting bottleneck for future generations of computing systems, as more digital logic and circuitry components are integrated onto a single die. Dynamic power management (DPM) techniques such as dynamic voltage and frequency scaling (DVFS) and power gating, are widely used in state-of-the-art processor systems to save power and improve energy efficiency. For example, Enhanced Intel SpeedStep Technology (EIST) [1], AMD PowerNow! [2], ARM Intelligent Energy Controller (IEC) [3] and NVIDIA Power Management Mode [4] allow the voltage and frequency (clock speed) of the processor to be dynamically changed (to different Power States) by software.

To employ a more effective power management strategy, many adaptive solutions have been developed recently by leveraging control theory and machine learning approaches. In these adaptive power management schemes, the control/learning agent can monitor workload status at run-time and adjust the voltage and frequency setting according to its online estimation model [5–8]. In conventional power delivery systems for multicore and manycore processors, a cluster of cores (or even all the cores) may reside in one voltage domain and share one voltage rail from an off-chip voltage regulator. Due to the long physical distance and associated parasitic loading effect, the voltage transition time of an off-chip voltage regulator generally exceeds a millisecond which fundamentally limits the rate at which power management settings can be adjusted in response to transient workload events, which can happen within several microseconds. Although the introduction of integrated voltage regulation allows a system to have much finer spatial (per-core) and temporal (tens to hundreds of nanoseconds) granularity of supply voltage allocation and delivery, [9, 10], a practical and effective method is still needed for adaptive power management at microsecond timescales.

In this paper, we propose F-LEMMA–a fast learning-based integrated voltage and frequency scaling approach for energy-efficient multicore and manycore processors. Leveraging novel power delivery systems with integrated voltage regulators, the processor supply voltage transition time drops to nanoseconds, which opens the door for microsecond timescale power management. We propose a hierarchical approach, where a global controller works as the interface to a userspace level energy and power management methodology; an intermediate learning-based controller takes in the architecture information and utilizes a reinforcement learning agent to update the configuration of a lower-level swift controller; finally, the swift controller uses a fast linear classifier to generate voltage and frequency pairs for each core at microsecond timescales. Experimental results show that F-LEMMA achieves a 35.2% energy saving on average across a wide range of benchmarks, at the cost of minimal overhead.

This paper makes the following contributions to the state of the art power management schemes:

- A comparison study of integrated voltage regulators and power delivery systems supporting microsecond timescale per-core fast DVFS with high power delivery efficiency.
- A hierarchical power management strategy including a global controller, a learning controller, and a swift controller, which leads to adaptive microsecond timescale per-core fast DVFS.

*Both authors contributed equally to this research.

MLCAD '20, November 16–20, 2020, Virtual Event, Iceland

ACM ISBN 978-1-4503-7519-1/20/11.
https://doi.org/10.1145/3380446.3430630

- A system implementation of the hierarchical power management approach and a High Level Synthesis (HLS) of its learning controller.
- A comprehensive experimental study of the proposed F-LEMMA approach, which demonstrates extra energy savings from fast power management compared to previous related work.

2 BACKGROUND

2.1 Dynamic Voltage Frequency Scaling (DVFS)

Dynamic voltage and frequency scaling (DVFS) is a technique to manage processor power consumption. Run-time dynamic power has a squared and linear relationship with frequency and voltage ($P_{\text{dynamic}} \sim CV^2f$), respectively, whereas static power has a linear relationship with voltage ($P_{\text{static}} \sim VI_{\text{static}}$).

Effective DVFS for multi-core processors requires multiple voltage domains. The circuitry within one voltage domain shares a common voltage rail, hence opportunities to reduce the domain's voltage are limited by the unit that needs the highest supply voltage. Voltage levels are scaled in fixed, discrete steps and are typically selected using tables that map frequency to voltage. Voltage and frequency scaling is based on the application's performance requirements. For example, when one core is waiting for synchronization with other cores, its voltage and frequency can be reduced to save power and energy.

2.2 Adpative Power Management

In recent years, adaptive power management has replaced previous fixed models for power management as the workloads in multicore and manycore systems have become more diverse and variable. Workloads are predicted at run-time using adaptive models for more effective power management. First, control theoretic mechanisms, such as Kalman filters [11] and model predictive control [12], use dynamically updated models to scale voltage and frequency under power or performance constraints.

Second, learning mechanisms predict application phases and control decisions without knowing an accurate workload model in advance. With reinforcement learning, an agent learns to act optimally in an environment by evaluating and selecting actions that optimize for desired rewards. Reinforcement learning can be adopted for power management by training a per-core DVFS agent that selects the appropriate voltage and frequency levels by observing system conditions [5, 7, 11]. As both adaptive control and the learning algorithm are relatively complex and take a long time to execute, such adaptive power management can only operate at low frequencies. In our work, this problem is addressed by introducing a hierarchical design with fast local controllers informed by a slower high-level learning-based management strategy.

2.3 Integrated Voltage Regulators

In a conventional power delivery system for multicore or even manycore processors, cores share a common voltage rail and a huge centralized voltage regulator is located off-chip to step down the supply voltage from the PCB board level (5-12V) to the core level (0.8-2V). As the off-chip voltage regulator uses large inductors and capacitors - and there are board level decoupling capacitors and noteworthy parasitic inductance - there is an unavoidable transition time (rise time and fall time) before the voltage reaches a desired level. Because of this large transition time, dynamic voltage and frequency scaling in processors with off-chip based power delivery systems is limited to millisecond timescales.

Emerging power delivery systems use integrated voltage regulators, moving the step-down voltage regulator on-chip. Integrated regulator design strives to reduce the size of inductors and capacitors for small on-die area. One prominent side effect of this design strategy is pushing the switching frequency from tens to hundreds of MHz. Such a higher switching frequency comes at a cost of more significant switching losses and degraded conversion efficiency.

The integrated voltage regulator naturally has a much shorter transition time than conventional off-chip voltage regulators [13–15]. This advantage comes from smaller inductors and capacitors, faster switching, and reduced parasitic inductance from its location closer to the core. Prototype designs [16–19] have shown that power delivery with integrated regulators can easily transition between voltage levels at nanosecond timescales.

2.4 Related Work

Although the general method of DVFS has been studied intensively in the past, only a subset of that work addresses the specific problem of adaptive voltage scaling at a faster time scale. Learning based predictors and controllers are proposed to find optimal power and performance in [5, 7, 20, 21]. Hierarchical power management [8, 22] has been widely adopted, from mobile devices [23] to cloud computers [24]. Limited by the supply voltage transition time in processor power delivery systems and the complexity of effective power management algorithms, these approaches operate at millisecond timescales. With the development of integrated voltage regulators, per-core microsecond level fast DVFS has become practical. Kim et al. [16], Toprak-Deniz et al. [17], Meinerzhagen et al. [25], Kim et al. [26], and Keller et al. [19] designed integrated voltage regulators that can support microsecond level dynamic voltage scaling. Kim et al. [9] studied the potential system level energy benefits from microsecond level dynamic voltage scaling supported by on-chip integrated voltage regulators. Höppner et al. [27] and Tseng et al. [18] studied fast DVFS on MPSoCs and SRAMs respectively. Kasture et al. [28] proposed a fine-grain DVFS scheme for latency-critical workloads. Bai et al. [29] proposed a voltage regulator efficiency aware power management strategy, which relies on reinforcement learning. Although the fast per-core DVFS supported by integrated voltage regulators offers a potential means to improve power and energy efficiency, effective power management strategies remain missing.

3 METHODOLOGY

3.1 F-LEMMA Framework

DVFS control algorithms can be implemented in the processor microarchitecture, in the scheduler, or through compiler algorithms. Most prior research in DVFS control is implemented in the operating system with coarse temporal granularity, a sensible approach when off-chip regulators have slow response times and voltages change on the order of several milliseconds. Integrated voltage regulators enable more responsive DVFS, saving power and energy at microsecond granularity, but effective mechanisms are required to guide such fine-grained DVFS.

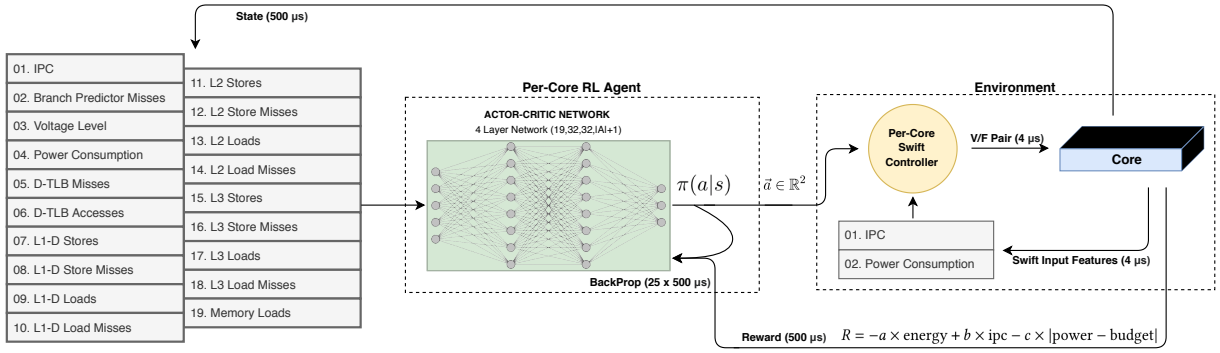

Figure 1: The proposed F-LEMMA power management framework with reinforcement learning and swift controllers

We propose a hierarchical DVFS management approach with three control layers. First, a global controller in user space specifies the power budget and energy-performance weights. Second, a per-core learning controller in the microarchitectural layer uses reinforcement learning and passes information to a swift controller. Third, a per-core swift controller in the circuit layer changes the core's voltages and frequencies at microsecond timescales.

3.2 Global Controller

The global controller provides a programmable interface that permits the operating system to specify energy and power management policies for service quality (QoS), application power management, and power capping. By default, the cores' energy and power budgets correspond to peak processor power. In contrast, under active and intelligent power management, Core(i,j) executing task i will be given budget $PB_{(i)}$.

$$PB_{(i)}(t) = \sum_{0}^{n_i} PB_{Core(i,j)}(t) = \alpha(t)(n_i/N)P_{processor} \quad (1)$$

where α is the programmed power allocation ratio and $P_{processor}$ is the processor peak power.

3.3 Learning Controller

Reinforcement Learning (RL) is a subset of machine learning built upon a Markov Decision Process, which describes interactions between an agent and its environment over time. The environment is represented by states. At each time step, the agent selects an action that changes the environment and thus the state. After selecting this action, the agent transitions to a new state and receives a reward associated with this state.

Table 1 lists components of an RL model. Note that actions or elements of the state space can be either continuous or discrete. The value functions describe the expected return for being in some state or for taking an action in a state when following policy π:

$$V(s) = \mathbb{E}_\pi\,[G|S_t = s] \quad (2)$$

$$Q(s,a) = \mathbb{E}_\pi\,[G|S_t = s, A_t = a] \quad (3)$$

The agent's goal is to learn policy $\pi(a|s)^*$, which maps each state to an action that maximizes the expected return G over k time-steps when future time-steps are discounted by factor γ^k.

Policy gradient methods, such as Actor-Critic, directly optimize the policy by approximating the policy π and value function (Q or V) using approximators such as neural networks [30]. The *actor* consumes the state and produces a probability distribution over the action space. The *critic* learns a real-valued number that approximates the value function, $V(s)$.

Table 1: RL terminology. RL's goal is to an find optimal policy $\pi(a|s)^*$

Terminology	Symbol
Action Space	$a \in A$
State Space	$s \in S$
Reward Function	$R \in \mathbb{R}^1$
Return	$G = \sum_{t=0}^{k} \gamma^k R_k$
Policy	$\pi(a|s)^*$
State Value Function	$V(s)$
State-Action Value Function	$Q(s,a)$

If the action space is discrete with size $|A|$, the final layer in the approximator network is a flattened vector of size $|A|$ that is passed through a softmax layer to produce a discrete probability distribution. If the action space is continuous (*e.g.*, $a \in [0,1]$), the final layer approximates a probability distribution by predicting its parameters. For example, a layer that approximates a normal distribution must predict the mean μ and variance σ of a, increasing the number of outputs required for the approximation function.

In each time step, the actor takes a normalized state as input, forward propagates the neural network approximator, and selects an action by sampling from the output distribution. A trajectory is built by saving actions, states, and rewards over several time steps. The actor and critic networks share weights and are trained jointly by back propagating with the appropriate loss functions and utilizing stored trajectories.

For power management, the environment is processor core activity and the state space is defined by 19 normalized performance counters including instruction throughput, branch prediction misses, cache misses, and reads as well as the current power and voltage levels. See Fig. 1 for details.

The reward function is a linear combination of instruction throughput, energy, and the power budget determined by the global controller [29].

$$R = -a \times \text{energy} + b \times \text{ipc} - c \times |\text{power} - \text{budget}| \quad (4)$$

where a, b, and c are experimentally determined parameters that ensure values are in a comparable range ($a \sim 1 \times 10^6$, $b \sim 1$, and

1 Learning Controller (with Swift) ($\sim 500\mu s$)

Input: $N_{\text{cores}}, f(s;\theta_1), \cdots, f(s;\theta_{N-1}), \hat{s}_{\text{mean}}, \hat{s}_{\text{std}}$
  $i \leftarrow 0$
  while $i < N_{\text{cores}}$ **do**
    $s \leftarrow \text{getCoreState}(i)$
    $s \leftarrow (s - \hat{s}_{\text{mean}}) / \hat{s}_{\text{std}}$
    $\mu_{\text{policy}}, \sigma_{\text{policy}}, V(s) \leftarrow f(s;\theta_i)$ (forward propagation)
    construct $\pi(a|s) \leftarrow \mathcal{N}\left(\mu_{\text{policy}}, \sigma_{\text{policy}}\right)$
    sample weights $\vec{w}_i \sim \pi(a|s)$
    updateSwiftController($i, \vec{w}_i$)
    $R \leftarrow \text{observeReward}(i)$
    store $\mu_{\text{policy}}, \sigma_{\text{policy}}, R, V(s)$
    $i \leftarrow i + 1$
  end while

$c \sim 1 \times 10^{-1}$). The influence of the global controller's budget is determined by parameter c. When $c = 0$, the global controller is effectively turned off.

The learning controller can manage the DVFS settings either independently or in coordination with the swift controller at a lower level. During independent management, it directly maps the core's state to a voltage-frequency pair. During coordinated management, it sends an intermediate weight vector to the swift controller as described in Algorithm 1.

3.4 Swift Controller

Each swift controller is associated with a core, managing its power and energy consumption by adjusting its voltage and frequency at microsecond timescales, which is supported by the integrated voltage regulator. First, the swift controller monitors current drawn by its core during each fine-grained monitoring interval (*e.g.*, 100ns) to calculate power consumption. Second, it accesses hardware performance counters. These measurements together influence voltage and frequency settings at microsecond timescales.

The swift controller uses a linear classifier as described in Eq. 5, where X is the input feature vector, a is the weight vector for the input feature, and b is the bias. When $f(X, a, b)$ is greater than threshold R_i, the swift controller sets voltage and frequency to V_i and F_i.

$$f(X, a, b) = Wa + b \tag{5}$$

Operating at microsecond timescales, the linear classifier must be computationally simple yet effective. The classifier takes only two run-time parameters, the power consumption and the instruction throughput, defining input $X = [P(t), IPC(t)]$. Beyond instruction throughput, we consider and test other performance counters such as cache hits and misses. At millisecond timescales, these counters improve model accuracy when estimating system dynamics. However, at the microsecond timescales we consider, these counters exhibit rapid and large fluctuations that can cause the system to oscillate and fail to converge.

We propose a hierarchical management strategy in which the global and learning controllers dynamically update the weight vector a. Updated weights help the swift controller capture diverse workload phases and variations. Depending on workload phase, power and IPC have different roles in estimating system behavior. For example, suppose the fixed-point unit dissipates less power and the floating-point unit dissipates more power. As a workload performs a varying mix of fixed and floating-point operations, simply using power or instruction throughput alone cannot accurately classify system behavior. Even a static combination of power and throughput may not be accurate. The pursuit of accurate classifiers is further complicated by other types of instructions in real multicore and manycore systems.

4 SYSTEM IMPLEMENTATION

In our design, the learning controller is executed on a dedicated digital logic controller located close to the core. Compared with execution at the software level of general-purpose CPUs, execution with dedicated hardware controller avoids redundancy from middleware, which increases both performance and energy efficiency. We compare the learning controllers for both software and hardware controller designs. In software, the learning controller takes on average 40 microseconds to execute on a 2.3 GHz Dual-Core Intel Core i5 processor. To estimate performance of the proposed hardware controller design, we synthesize the learning controller on a Zynq-7000 FPGA using Xilinx Vivado HLS 2019.1, with a pipeline applied to optimize the design. The learning controller takes 1464 cycles to execute, with an average power consumption of 1.38 W when active. Scaled to processor-level technology and design, this online learning controller can execute within 10 microseconds and introduces less than 2% overhead with a lower power consumption. The hardware controller design thus speeds up the learning controller by 4 times versus execution in software. The swift controllers operate at microsecond timescales. Each swift controller operation has 2 fixed point multiplications, 1 addition, and up to 3 comparisons. The overheads from the swift controllers are negligible compared with those from the learning controller.

5 EVALUATION RESULTS

5.1 System Setup

We evaluate the proposed hierarchical learning-based power management scheme with experiments on an Intel Nehalem x86 processor, which is detailed in Table 2. We use Sniper v7.3 [31] (with Mcpat [32]) to simulate system performance and power for this manycore processor, generating run-time statistics with a granularity of 100 ns. We integrate both Numpy and PyTorch packages with

Table 2: Architecture parameters and hyperparameters for the hierarchical controller.

Configurations	Value
Number of cores	2-128
Core architecture	Intel Nehalem (x86)
V/F Levels (V/GHz)	1.20/2.0,1.08/1.8,0.96/1.6,0.84/1.4
Nominal V/F	1.20/2.0
L1-I/D cache	32KB, 4-way, LRU
L2 cache	512KB, 8-way, LRU
L3 cache	8MB, 16-way, LRU
Global/learning/swift ctrl.	10 ms, $500\mu s$, $4\mu s$
NN Architecture	4-layer (19,32,32,$\lvert A\rvert$ + 1)
Learning rate	1×10^{-3}
Discount reward factor	$\gamma = 0.95$
Trajectory size for backprop	25
Optimizer	Adam ($\beta_{1,2} = 0.9, 0.999$)
Technology node	45nm

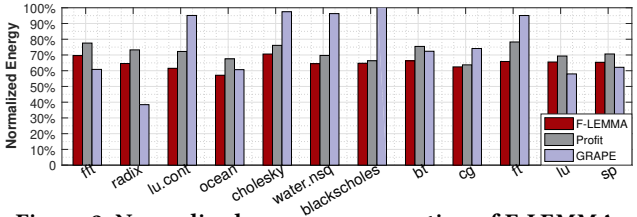

Figure 2: Normalized energy consumption of F-LEMMA

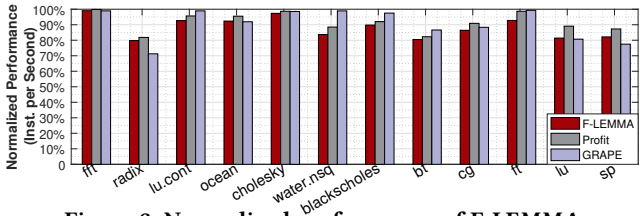

Figure 3: Normalized performance of F-LEMMA

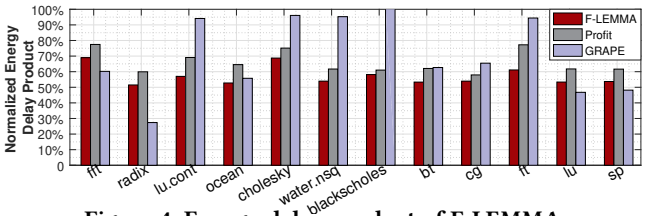

Figure 4: Energy delay product of F-LEMMA

Sniper in order to implement the hierarchical design. Sniper performs timing simulations for multithreaded, shared-memory applications with tens to hundreds of cores, and has been validated for Intel Core2 and Nehalem systems. We select representative benchmarks that cover a wide range of scientific and computational domains from the *parsec*, *splash2*, and *NPB* benchmark suites. The global controller is triggered by user space power management. The learning controllers operate every 500 microseconds, a rate limited by the computational complexity of the learning algorithm. The swift controllers work at 4 microsecond scales, as determined by the voltage transition times of the integrated voltage regulators.

5.2 Performance of the Learning Controller

We compare F-LEMMA to the following techniques. For fairness, F-LEMMA (learning controller), Profit, and GRAPE all operate at a fixed timescale of 500 microseconds within Sniper.

- **Default Race-to-Idle.** Runs each benchmark as fast as possible. All other methodologies are normalized to this.
- **Profit Style.** State-of-the-art *reinforcement learning based* power, and energy management for multicore and manycore systems [5].
- **GRAPE Style.** State-of-the-art *feedback control based* power and energy management for multicore and manycore systems with performance constraints [11].

We evaluate *energy consumption*, not power dissipation, for a standard comparison against workloads and configurations. The energy metric evaluates net benefits and accounts for potential losses due to extended execution time when lowering frequency. We normalize energy results to the Default Race-to-Idle.

Fig. 2 shows the normalized energy consumption and Fig. 3 shows the normalized performance loss (instructions per second). F-LEMMA achieves 35.2% energy saving with an 11.8% performance penalty, on average, compared to Default Race-to-Idle. The best case is the *fft* benchmark which saves 30.4% energy with only a 1.0% performance loss. The worst is the *radix* benchmark which saves 30.4% energy at a 25.3% performance loss.

Compared to Profit and GRAPE, F-LEMMA achieves 6.6% and 11.5% extra energy saving with 3.5% and 2.6% performance penalty. For the *fft*, *lu.cont*, *cholesky*, *water.nsq*, *blackscholes* and *ft* benchmarks, DVFS saves significant amounts of energy with minimal performance penalty across all three power management approaches.

Fig. 4 shows the *Energy-Delay Product* normalized to Default Race-to-Idle. Across most benchmarks, F-LEMMA has the highest energy efficiency and smallest energy-delay product, after accounting for potential performance losses. On average, F-LEMMA, Profit and GRAPE have normalized energy-delay products of 0.73, 0.78, and 0.84, respectively.

5.3 Performance of Each F-LEMMA Layer

Figures 5–6 compare the energy savings and performance penalties from F-LEMMA and alternatives that use only a subset of the global, learning, and swift controllers. F-LEMMA outperforms a framework with only global and learning controllers (*i.e.*, the second bar), achieving significant energy savings with only a tiny performance loss. For example, on the *lu.cont*, *ocean*, and *ft* benchmarks, F-LEMMA achieves 9%, 8% and 6% energy saving respectively, while reducing performance by less than 1%. F-LEMMA also outperforms a framework with only the swift controller (*i.e.*, the third bar).

We also compare full hierarchical management with different configurations at each layer. Suppose the learning controller only pursues energy savings because the global controller specifies weights (1,0,0) for its reward function (*i.e.*, the fourth bar). The system achieves more energy saving but with slightly greater performance penalties. Finally, suppose the swift controller uses only power as the input feature and neglects instruction throughput (*i.e.*, the fifth bar). Compared to F-LEMMA, this configuration induces larger performance penalties for the same energy savings. With only power measurements, the swift controller predicts the effects of DVFS less accurately. These effects were discusssed in Section 3.3.

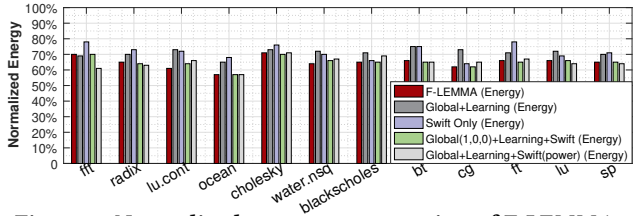

Figure 5: Normalized energy consumption of F-LEMMA

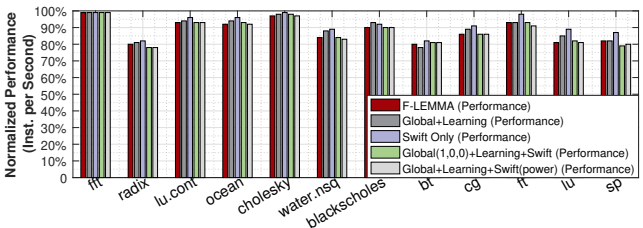

Figure 6: Normalized performance of F-LEMMA

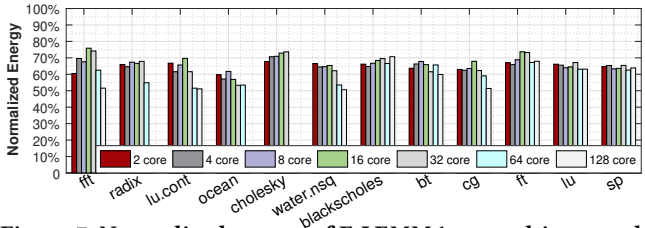

Figure 7: Normalized energy of F-LEMMA on multicore and manycore processors

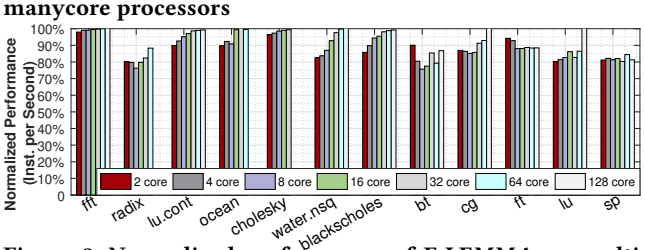

Figure 8: Normalized performance of F-LEMMA on multicore and manycore processors

5.4 Scalability on Manycore Systems

Fig. 7 and Fig. 8 show the energy and performance when scaling from 2 cores to 128 cores; some bars are blank because the benchmark does not support that configuration. Overall, as the number of cores scales from 2 to 128, F-LEMMA achieves 35-42% energy savings at a cost of 5-12% performance penalties. As the number of cores increases, the performance penalty decreases as more DVFS opportunities are created by more thread synchronizations.

6 CONCLUSION

In this paper, we proposed F-LEMMA, a hierarchical fast integrated voltage and frequency scaling approach for multicore and manycore processors. With integrated voltage regulators, DVFS power management can reach microsecond timescales. A learning based hierarchical approach, including a global controller in userspace, a learning controller at the architecture level, and swift controllers at the digital circuit level, is presented to guide microsecond level power management. Experimental results show that on average F-LEMMA can save 35.2% of energy with a 11.8% performance decrease. Compared with two classic millisecond timescale DVFS techniques using control theory and reinforcement learning, the F-LEMMA achieves 5% and 11% EDP improvements, respectively.

ACKNOWLEDGMENT

The research described in this paper was partly supported by Semiconductor Research Corporation (SRC) task 2810 and task 2821; NSF grant CNS-1739643, NSF grant CNS-1822085, NSF grant CCF-1527610, and NSF grant CCF-1408784; Samsung Research; and Lenovo Research. We are also grateful to the reviewers for their feedback.

REFERENCES

[1] Wikipedia. Speedstep. [EB/OL]. https://en.wikipedia.org/wiki/SpeedStep/.
[2] AMD Staff. Amd powernow! technology brief. *Advanced Micro Devices, Inc.*
[3] Teodor Neagoe, Ernest Karjala, and Logica Banica. Why arm processors are the best choice for embedded low-power applications? In *2010 IEEE 16th International Symposium for Design and Technology in Electronic Packaging (SIITME)*.
[4] The FPS Review. Nvidia power. [EB/OL]. https://www.thefpsreview.com/2019/12/04/nvidia-geforce-driver-power-mode-settings-compared/.
[5] Zhuo Chen, Dimitrios Stamoulis, and Diana Marculescu. Profit: priority and power/performance optimization for many-core systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2017.
[6] Muhammad Shafique, Benjamin Vogel, and Jörg Henkel. Self-adaptive hybrid dynamic power management for many-core systems. In *DATE*. IEEE, 2013.
[7] Hwisung Jung and Massoud Pedram. Supervised learning based power management for multicore processors. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 29(9):1395–1408, 2010.
[8] Zhiyuan Ren, Bruce H Krogh, and Radu Marculescu. Hierarchical adaptive dynamic power management. *IEEE Transactions on Computers*, 54(4):409–420, 2005.
[9] Wonyoung Kim, Meeta S Gupta, Gu-Yeon Wei, and David Brooks. System level analysis of fast, per-core dvfs using on-chip switching regulators. In *HPCA*, 2008.
[10] Brian Zimmer, Yunsup Lee, Alberto Puggelli, Jaehwa Kwak, Ruzica Jevtić, Ben Keller, Steven Bailey, Milovan Blagojević, Pi-Feng Chiu, Hanh-Phuc Le, et al. A risc-v vector processor with simultaneous-switching switched-capacitor dc–dc converters in 28 nm fdsoi. *IEEE Journal of Solid-State Circuits*, 2016.
[11] Muhammad Husni Santriaji and Henry Hoffmann. Grape: Minimizing energy for gpu applications with performance requirements. In *2016 IEEE/ACM MICRO*.
[12] Abhinandan Majumdar, Leonardo Piga, Indrani Paul, Joseph L Greathouse, Wei Huang, and David H Albonesi. Dynamic gpgpu power management using adaptive model predictive control. In *HPCA*. IEEE, 2017.
[13] An Zou, Jingwen Leng, Yazhou Zu, Tao Tong, Vijay Janapa Reddi, David Brooks, Gu-Yeon Wei, and Xuan Zhang. Ivory: Early-stage design space exploration tool for integrated voltage regulators. In *Proceedings of the 54th DAC 2017*.
[14] An Zou, Jingwen Leng, Xin He, Yazhou Zu, Christopher D Gill, Vijay Janapa Reddi, and Xuan Zhang. Voltage-stacked gpus: A control theory driven cross-layer solution for practical voltage stacking in gpus. In *2018 51st Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE.
[15] An Zou, Jingwen Leng, Xin He, Yazhou Zu, Vijay Janapa Reddi, and Xuan Zhang. Efficient and reliable power delivery in voltage-stacked manycore system with hybrid charge-recycling regulators. In *2018 55th ACM/ESDA/IEEE DAC*. IEEE.
[16] Wonyoung Kim, David M Brooks, and Gu-Yeon Wei. A fully-integrated 3-level dc/dc converter for nanosecond-scale dvs with fast shunt regulation. In *ISSCC*. IEEE, 2011.
[17] Zeynep Toprak-Deniz, Michael Sperling, John Bulzacchelli, Gregory Still, Ryan Kruse, Seongwon Kim, David Boerstler, Tilman Gloekler, Raphael Robertazzi, et al. 5.2 distributed system of digitally controlled microregulators enabling per-core dvfs for the power8 tm microprocessor. In *ISSCC*. IEEE, 2014.
[18] 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. *IEEE journal of solid-state circuits*, 45(11):2239–2249, 2010.
[19] Ben Keller, Martin Cochet, Brian Zimmer, Yunsup Lee, Milovan Blagojevic, Jaehwa Kwak, Alberto Puggelli, Stevo Bailey, Pi-Feng Chiu, Palmer Dabbelt, et al. Sub-microsecond adaptive voltage scaling in a 28nm fd-soi processor soc. In *ESSCIRC*. IEEE, 2016.
[20] Hao Shen, Jun Lu, and Qinru Qiu. Learning based dvfs for simultaneous temperature, performance and energy management. In *ISQED*. IEEE, 2012.
[21] Martin Rapp, Anuj Pathania, Tulika Mitra, and Jörg Henkel. Prediction-based task migration on s-nuca many-cores. In *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1579–1582. IEEE, 2019.
[22] Thannirmalai Somu Muthukaruppan, Mihai Pricopi, Vanchinathan Venkataramani, Tulika Mitra, and Sanjay Vishin. Hierarchical power management for asymmetric multi-core in dark silicon era. In *DAC*. IEEE, 2013.
[23] Jacob Sorber, Nilanjan Banerjee, Mark D Corner, and Sami Rollins. Turducken: hierarchical power management for mobile devices. In *MobiSys*, 2005.
[24] Nikolas Ioannou, Michael Kauschke, Matthias Gries, and Marcelo Cintra. Phase-based application-driven hierarchical power management on the single-chip cloud computer. In *PACT*. IEEE, 2011.
[25] Pascal Meinerzhagen, Carlos Tokunaga, Andres Malavasi, Vaibhav Vaidya, Ashwin Mendon, Deepak Mathaikutty, Jaydeep Kulkarni, Charles Augustine, Minki Cho, Stephen Kim, et al. An energy-efficient graphics processor featuring fine-grain dvfs with integrated voltage regulators, execution-unit turbo, and retentive sleep in 14nm tri-gate cmos. In *ISSCC*. IEEE, 2018.
[26] Stephen T Kim, Yi-Chun Shih, Kaushik Mazumdar, Rinkle Jain, Joseph F Ryan, Carlos Tokunaga, Charles Augustine, Jaydeep P Kulkarni, Krishnan Ravichandran, James W Tschanz, et al. Enabling wide autonomous dvfs in a 22 nm graphics execution core using a digitally controlled fully integrated voltage regulator. *IEEE Journal of Solid-State Circuits*, 51(1):18–30, 2015.
[27] Sebastian Höppner, Chenming Shao, Holger Eisenreich, Georg Ellguth, Mario Ander, and René Schüffny. A power management architecture for fast per-core dvfs in heterogeneous mpsocs. In *ISCAS*. IEEE, 2012.
[28] Harshad Kasture, Davide B Bartolini, Nathan Beckmann, and Daniel Sanchez. Rubik: Fast analytical power management for latency-critical systems. In *MICRO*. IEEE, 2015.
[29] Yuxin Bai, Victor W Lee, and Engin Ipek. Voltage regulator efficiency aware power management. In *ISPLOS*, 2017.
[30] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA, 2018.
[31] Trevor E Carlson, Wim Heirman, and Lieven Eeckhout. Sniper: Exploring the level of abstraction for scalable and accurate parallel multi-core simulation. In *SC*, 2011.
[32] 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 *MICRO*, 2009.