Application and Thermal-reliability-aware Reinforcement Learning Based Multi-core Power Management

SAI MANOJ PUDUKOTAI DINAKARRAO, George Mason University, USA
ARUN JOSEPH and ANAND HARIDASS, IBM Systems, India
MUHAMMAD SHAFIQUE, Vienna University of Technology (TU Wien), Austria
JÖRG HENKEL, Karlsruhe Institute of Technology, Germany
HOUMAN HOMAYOUN, University of California, Davis, USA

Power management through dynamic voltage and frequency scaling (DVFS) is one of the most widely adopted techniques. However, it impacts application reliability (due to soft errors, circuit aging, and deadline misses). However, increased power density impacts the thermal reliability of the chip, sometimes leading to permanent failure. To balance both application- and thermal-reliability along with achieving power savings and maintaining performance, we propose application- and thermal-reliability-aware reinforcement learning–based multi-core power management in this work. The proposed power management scheme employs a reinforcement learner to consider the power savings and variations in the application and thermal reliability caused by DVFS. To overcome the computational overhead, the power management decisions are determined at the application-level rather than per-core or system-level granularity. Experimental evaluation of proposed multi-core power management on a microprocessor with up to 32 cores, running PARSEC applications, was done to demonstrate the applicability and efficiency of the proposed technique. Compared to the existing state-of-the-art techniques, the proposed technique enables an average energy savings of up to ~20%, up to 4.926 °C temperature reduction without degradation in the application- and thermal-reliability.

CCS Concepts: • **Hardware → On-chip resource management**; **Chip-level power issues**; *Temperature optimization*; *Transient errors and upsets*; Process, voltage and temperature variations;

Additional Key Words and Phrases: Multi-core processor, reinforcement learning, application reliability, thermal reliability, power management, DVFS

1 INTRODUCTION

The ever-increasing proliferation of multi-core processors into the computing systems (ranging from portable devices to datacenters) facilitate the multi-program execution of multi-threaded applications. This enables high performance under tight power budgets (Bergamaschi et al. 2008; Manoj et al. 2015, 2017; Pagani et al. 2017; Pagani et al. 2018; Tarsa et al. 2014; Wang and Pedram 2016). The high performance with multi-core systems coupled with increased power density poses multiple challenges, with reliability being one of the key design parameters to be considered along with power/energy and performance across a wide range of computing platforms, from miniature embedded systems to massive data centers (Shafique et al. 2014; Swaminathan et al. 2017). What is more, the increased power consumption forms a positive loop with temperature leading to increased temperatures, eventually leading to thermal runaway failures (Wu et al. 2014; Xu et al. 2015). To overcome such concerns, reliability-aware power management is critical for processors embedded in small-scale systems as well as in datacenters. Here, the term *"reliability" encompasses both application reliability and thermal reliability. Application reliability* is composed of two parts: (i) functional reliability, i.e., for a given input, the correctness of output values of a given function considering faults such as soft errors in the underlying hardware; and (ii) timing reliability, i.e., the ability to meet the timing requirements. Though the *thermal reliability* is dependent on multiple factors, we consider the predominant factors, oxide breakdown and the electron migration (Gnad et al. 2015; Manoj et al. 2013; Pagani et al. 2014; Srinivasan et al. 2004), in this work.

Towards optimizing and meeting the power budget constraints, Dynamic Voltage and Frequency Scaling (DVFS) (Esmaeilzadeh et al. 2011; Manoj et al. 2015; Pagani et al. 2017; Pagani et al. 2018; Tarsa et al. 2014; Wang and Pedram 2016) has proven to be one of the most effective and widely used techniques with adaptivity for power/energy savings. In the former works, DVFS is performed considering different parameters such as worst-case execution time of the task (Choi et al. 2005), temperature (Lee et al. 2010), and voltage demand (Choi et al. 2004; Dietrich et al. 2010). Many of the existing works, such as Choi et al. (2005), Dietrich et al. (2010), and Wang and Pedram (2016), perform DVFS by predicting one or more parameters for the next time interval(s). Based on this, the VF settings are applied accordingly towards meeting the power/energy budgets under the constraints of performance requirements. Advancements in the machine learning (ML) field led to its adoption for prediction and/or on-chip parameter adaptations required for power management (DVFS) using techniques such as Bayesian learning (Wang et al. 2011), reinforcement learning (Jung and Pedram 2010; Manoj et al. 2016; Shen et al. 2013), and regression analysis (Bartolini et al. 2013; Bartolini et al. 2011; Manoj et al. 2018; Yang et al. 2015).

In addition to the power constraints, the on-chip temperature is one of the major concerns in multi-core processors that can have non-trivial impact on the lifetime and reliability of the chip (Shafique et al. 2014). Keeping the chip's temperature under a certain thermal threshold (or critical value) is of paramount importance, as otherwise high temperatures may cause permanent failures. To achieve this, i.e., to dissipate the heat and reduce the temperature, chips are provided with a cooling solution (e.g., the coupling of the thermal paste, heat spreader, heat sink, and cooling fan). It needs to be noted that power management aids to reduce the on-chip hot-spots, as the heat is generated from the consumed power. However, the power management primarily focuses on optimizing the power, and persistent consumption of power (even if it is low) leads to hot-spots, which might not be mitigated with power/energy saving–oriented DVFS techniques. To provide a better temperature regulation, the multi-core systems are equipped with Dynamic Thermal Management (DTM) technique. These DTM techniques are commonly reactive (i.e., triggered once the critical temperature is exceeded) and can power-down cores, reduce their supply voltages and execution frequencies, gate their clocks, boost the fan speed, and so on. In other words, if the chip heats up
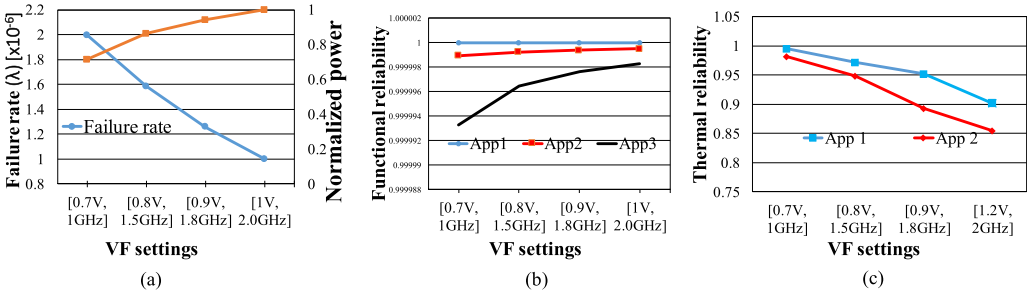
Fig. 1. (a) Fault rate and power consumption (b) application reliability and (c) thermal reliability under different VF settings.

above a critical value (identified using thermal sensors distributed across the chip), then the DTM is triggered to reduce the temperature. Similar to power management, machine learning is widely deployed for thermal management as well. Techniques for thermal management with machine learning, such as DTM with temperature prediction by regression (Lee et al. 2010), Q-learning (Lu et al. 2015; Shen et al. 2012), and so on, are proposed in the literature.

Many of the existing power management and thermal management works primarily focus on either optimizing the power and/or temperature of the system. Despite the power/thermal management optimizing the power and temperature of the chip, it highly degrades the reliability of the system components such as processor core, application data (cache), and memories, especially in the scaled geometries, resulting in induction of faults in the data (Makhzan et al. 2007; Sasan et al. 2009). The state-of-the-art soft-error reduction techniques mainly exploit software- and hardware-level techniques (Kapadia and Pasricha 2015; Mukherjee et al. 2002; Qi et al. 2010; Shye et al. 2007; Xu et al. 2013). However, these techniques are computationally expensive due to the continuous redundancy checking happening at software-levels. Further, reliability-aware techniques with power optimization, such as Dabiri et al. (2007), and Wu and Marculescu (2014), require technology node changes such as transistor sizing. Though such techniques can achieve desired reliability, they demand excessive design and manufacturing efforts; also, the reliability is more affine towards the soft-errors, and not considering the physical reliability. To observe the impact of DVFS, i.e., VF scaling on application and thermal reliability, and to determine the need of performing both application and thermal reliability–based power management, a motivational case study is carried out and presented below.

1.1 Motivational Case Study

A simple case study to understand the impact of VF scaling on the application and thermal reliability is presented in Figure 1. The model based on which the application and thermal reliability are derived is presented in Section 2.3, and the experimental settings are described in Section 6. As can be observed from Figure 1(a), with scaling down of voltage-frequency levels, the power consumption decreases, but the fault-rate increases. Also, from Figure 1(b) it can be seen that the reliability for different applications is different, even under the same VF settings. Thus, *an application's reliability is not a simple function of VF, rather it is a function of application characteristics such as runtime and instruction profile.* Similar findings have been reported in Salehi et al. (2015) and Salehi et al. (2015). It needs to be noted that the plotted functional reliability is under the best settings, i.e., minimal failure rate and deadline misses. To address this problem, in the proposed reinforcement learning–based power management, reliability is learned and considered as a feedback for the DVFS, along with the achieved power saving.

In addition to application reliability, the thermal reliability w.r.t. VF scaling is shown in Figure 1(c) for different PARSEC benchmark applications. Here, the power and temperature values are obtained from McPAT (Li et al. 2009) and HotSpot (Huang et al. 2006). The simulations are run in SniperSim (Carlson et al. 2014) with more details presented in Section 6. As one can observe from Figure 1(c), with the scaling up of VF levels, thermal reliability decreases, i.e., power consumption and temperature increases leading to reduced thermal reliability. However, the application reliability increases and vice versa. It has to be noted that the application and thermal reliabilities of different applications are different even under same VF settings, due to their inherent characteristics. As such, an optimal DVFS that meets the power-performance budget without degrading the thermal and application reliabilities is needed. As the power management focused works can lead to degradation of the reliability of the application and system, the power management under the constraints of reliability is non-trivial. Thus, **the objective of this work is to perform power management under the constraints of performance, application, and thermal reliability, i.e., to achieve low power consumption along with meeting the reliability constraints and desired performance**.

Associated Research Challenges

The associated challenges of paramount importance to perform learning-based power management, considering the application characteristics and reliability, can be outlined as follows:

Computational Overheads: The power management can be performed at different levels of abstraction such as at core-level or system-level. Performing per-core power management introduces computational and hardware overheads such as VF controller per-core (Jung and Pedram 2010; Manoj et al. 2015, 2018; Shafique et al. 2016). However, system-level power management refers to performing power management at a granularity of system-level, which is efficient in terms of computational overhead, but achieves lower power savings and/or energy efficiency (Shen et al. 2013). Use of VF-island-based power management, though efficient, lacks flexibility and scalability (Rangan et al. 2009). As such, an intermediate solution is desired.

Application-Reliability Variation with DVFS: In addition to the traditional power management challenges such as processing overhead, embedding reliability for power management adds the following challenges: the reliability of an application varies with the VF levels at which the application is being executed and also the reliability for different applications is different (Figure 1). Additionally, to learn the reliability of an unseen application for an efficient power management, the supervised learning is not an effective solution, as the reliability is hard to predict pro-actively or known *a priori* for unseen applications (Wang and Pedram 2016).

Thermal Reliability with Power Consumption: As mentioned earlier, the power consumption leads to heat generation and can lead to thermal hot-spots on-chip eventually causing permanent failures. Power consumption–based reduction or performing DVFS to lower/mitigate the hot-spots leads to reduced performance as well as affecting the application reliability. In addition, thermal reliability is inversely proportional to the on-chip temperature, i.e., the higher the temperature, the lower the thermal reliability. Low temperature arises from lower power consumption, which implies that improving thermal reliability has inverse effects on application reliability. As such, a trade-off has to be maintained between thermal and application reliability.

Contributions of This Work

To address the above-discussed problems, in this article, we make the following novel contributions:

- *To the best of our knowledge, this is the first work that considers both application and thermal reliability along with performance to perform multi-core power management.*

- To achieve desired application and thermal reliabilites along with power/energy savings, a reinforcement learning (RL)–based power manager is proposed. Here, the RL agent determines the VF levelbased on the predicted power and the achieved reliability.
- The reward is determined based on the power savings, and temperature, and application reliability, which allows the power manager to optimize the power and temperature while maintaining the application and thermal reliability.

Traditional power management works consider power-performance trade-off and do emphasize reliability concerns. Similarly, reliability-aware works are limited to either power optimization or concerned about one kind of reliability enhancement. In contrast, this work considers both thermal and application reliability compared to existing works. Furthermore, as reliability cannot be aforementioned, this work is one of the first to utilize machine learning to adapt to variations in reliability during runtime.

Paper Organization

The rest of the paper is structured as follows: The models for the system, reliability, and applications employed in this work are presented in Section 2. The system architecture is discussed in Section 3. An introduction to reinforcement learning is presented in Section 4. Section 5 describes the proposed reliability-aware power management scheme. Section 6 presents experimental evaluation and comparison of proposed reliability-aware power management with other state-of-the-art techniques. Conclusions are drawn in Section 7.

2 SYSTEM MODEL

2.1 Hardware Architecture Model

We consider a homogeneous multi-core processor comprising of N cores, $C = \{C_1, C_2, \ldots, C_N\}$. Due to varying workloads, different cores execute at different frequencies to ensure proper execution. There exists a maximum operating frequency level f_{max} for every possible operating voltage V. The frequencies of a core can be varied between f_{min} to f_{max}, and the corresponding voltages between v_{min} and v_{max}. The cores operating at higher VF levels consume more power when executing the application. Furthermore, similar to Salehi et al. (2015), we assume that performance of the processor core is higher when running at a higher VF level.

2.2 Application Model

We consider a mixture of single-threaded and multi-threaded applications in this work, and each core executes one thread. Figure 2 represents a snapshot of multi-core system with multiple applications deployed. In Figure 2, different shades on processor cores represent different applications running on them. The distribution of applications is not uniform, i.e., different applications can run on different number of cores, depending on the number of threads. Each of the applications are composed of multiple tasks. A task τ requires w clock-cycles for execution. Also, at any given time, the total number of executed threads are smaller or equivalent to number of cores, similar to Pagani et al. (2017).

2.3 Reliability Models

Here, we present the employed application and thermal reliability models, followed by the power model for the applications running on a multi-core system.

2.3.1 Application Reliability Model. To determine the application reliability model, we consider transient faults and the timing reliability model. Transient fault occurrences are assumed to follow
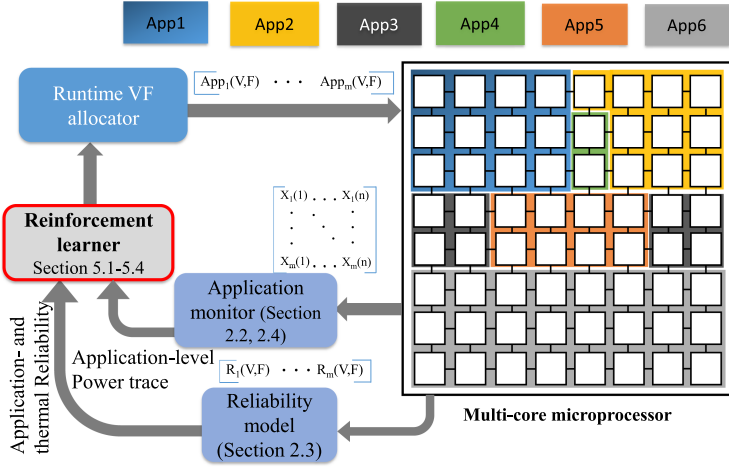
Fig. 2. Multi-core microprocessor equipped with the proposed application- and thermal-reliability-aware power manager.

a Poisson process with a rate of λ (Ejlali et al. 2012). The fault rate varies exponentially with the operating voltage (Zhu et al. 2004). As such, the transient fault rate, depending on the operating voltage V is

$$\lambda(V) = \lambda_0 10^{\frac{V_{max}-V}{\Delta}}, \tag{1}$$

where, λ_0 (= 10^{-6}) indicates the fault rate when operating at maximum possible voltage V_{max}; and Δ (= $1V$) is a parameter that indicates increase in fault rate when the voltage is decreased by one level. As the transient faults in the underlying hardware results in software faults, the Functional Vulnerability Index (FVI), as in Salehi et al. (2015), is considered, set to 1. The Functional Reliability (FR) model due to transient fault (λ) and software failure rate $\lambda(V) \times FVI$ is modeled as below:

$$FR(FVI, w, V, f) = e^{-\lambda(V) \times FVI \times \frac{w}{f}}, \tag{2}$$

where w indicates the number of clock-cycles needed to execute the application, and f represents the operating frequency. The employed reliability model is based on single task execution model, as in Ejlali et al. (2012). One of the main reasons to consider this model is that the adopted reliability models are shown to be accurate and robust for reliability estimation in Salehi et al. (2015) with <2.5% deviation in terms of reliability efficiency. However, it needs to be noted that the proposed power management scheme is independent of the fault model used, as the reward function requires the reliability variation rather than absolute reliability values. However, other application reliability models can be employed, as the proposed technique requires information regarding the reliability rather than the model information.

2.3.2 Thermal Reliability Model. The thermal reliability of the system depends on multiple factors, and oxide breakdown and electron migration (EM) are the predominant factors (Srinivasan et al. 2004). As such, the thermal reliability of the system is given by

$$R(t) = exp\left(-C \cdot t^\beta \cdot e^{-\frac{Ea\beta}{kT}}\right), \tag{3}$$

where $R(t)$ indicates the reliability at time instant t, $C = (\frac{1}{\Gamma(1+1/\beta) \cdot J^{-n}})^\beta$ with n is material based constant (1.1 for copper (Srinivasan et al. 2005)), J being the energy consumption, β is the Weibull

slope parameter (= 2, (Wu et al. 2002)), k is the Boltzman constant, E_a is the activation energy (0.9eV for copper).

Based on Equation (3), the reliability of a dual-core system having power consumption P_1 and P_2 leading to temperatures T_1 and T_2 is given as

$$R_2(t) = exp\left(-C \cdot t^{\beta} \cdot \left(e^{-\frac{E_a\beta}{kT_1}} + e^{-\frac{E_a\beta}{kT_2}} \right) \right). \tag{4}$$

In this work, the temperature (T_1, T_2) is obtained from HotSpot (Huang et al. 2006) and power of the cores are obtained from the McPAT (Li et al. 2009) directly.

2.4 Power Model

The total power consumption of a core is composed of static and dynamic power. The static power is dominantly due to leakage power and varies exponentially with threshold voltage. The dynamic power consumption is due to the application-dependent switching activities in the core. The total power consumption (Brooks et al. 2007; Ejlali et al. 2012) when operating at voltage V and frequency f is modeled as below:

$$P(V, f) = P_{static} + P_{Dynamic} = I_0 e^{\frac{-V_{th}}{\eta V_T}} V + \alpha C V^2 f. \tag{5}$$

Here, I_0 and η are technology parameters; V_T is the thermal voltage; V_{th} is the threshold voltage; α represents the switching activity factors, and C is the average capacitance. To obtain per-application power or energy trace, we sum the power traces of the cores on which the application is executing.

3 SYSTEM ARCHITECTURE

Figure 2 illustrates the system architecture with the proposed reliability-aware power management for a multi-core microprocessor. The microprocessor is composed of multiple cores running different applications on it. Each of the cores is equipped with private L1 and L2 caches. Characteristics such as per-application power trace (in mW) and the reliability are obtained or derived for the purpose of power management. The obtained application power trace and the derived reliability is fed to the RL-based power manager for generating the power management policy and to provide the optimal DVFS configuration.

The power management settings, i.e., VF levels, are determined in the OS layer. The power and reliability data obtained from the application logs are collected iteratively over a time-window of length n ($10\mu s$ in this work) is fed to the power manager to learn the power profile and derive reliability for different applications. The power manager determines the optimal power management policy based on the sensed data (power trace) and its reliability. The key advantage of employing a reinforcement technique is that the decision is learnt based on its experience rather than using labels that might prove to be less effective, especially considering reliability, which is different for different applications. Moreover, the decision made by the RL changes if the achieved reward is decreasing (or going in a negative direction), which facilitates to improve the quality of power management. To overcome the convergence constraints of the RL, the threshold on number of loops to be run is enforced, as use of deep RL might increase latency and operational costs. Furthermore, the power management is carried out at regular intervals (n) to facilitate sufficient time for switching activities and the decision making. More details on the simulation settings are provided in Section 6. It needs to be noted that the power and reliability data presented in Figure 2 are vectors and is a function of time. The application-level power trace is represented as a matrix X, where each column represents the power trace for different applications at one time instant. Similarly, the reliability is represented as vector R.

4 REINFORCEMENT LEARNING (RL)

Reinforcement learning (RL) is an ML technique that mimics one of the most common learning styles in natural life, i.e., to learn to achieve a goal by trial-and-error interaction with a dynamic or uncertain environment (Liu et al. 2010; Tan et al. 2009). In RL, interactions between the learning agent and the environment are generally modeled using a finite state space S (corresponding to environment inputs), a set of available actions A (corresponding to control/optimization knobs used by the agent), and a reward function $R : S \times A \rightarrow R$ (used to decide which action to take for a given state). The ultimate goal of RL is to figure out a policy $\pi(s) = a$, which chooses action $a \in A$ in each state $s \in S$ (i.e., a mapping between the states and the actions), to optimize a reward function (i.e., to maximize the cumulative rewards over a potentially infinite time span).

Q-learning: Q-learning is one of the most popular algorithms used to perform RL (Liu et al. 2010; Tan et al. 2009). In Q-learning, a Q-value is associated to every state-action pair (s, a), denoted as $Q(s, a)$. The value of $Q(s, a)$ approximates the expected long-term cumulative reward of taking action a starting from state s. In this way, the agent decides which action has to be performed in the current state to achieve the maximum long-term rewards based on the value function $Q(s, a)$. Namely, at decision epoch t_k when the system has just transitioned to state $s_k \in S$, the action a_k with the highest Q-value will be chosen. During the first few iterations, the RL chooses an action randomly; and based on the obtained reward, the actions are learnt. The Q-learning has the benefit: As it is a model-free learning algorithm, it is not necessary for the Q-learning agent to have any prior system information, such as the transition probability from one state to another. Therefore, it is a highly adaptive and flexible technique, which is one of the reasons it is considered in this work.

The fundamental aspect of Q-learning algorithm is the value iteration update of the Q-value function. Particularly, the Q-value for each state-action pair is initially pre-defined (or set randomly). However, these values are updated every time an action is issued and a reward is received. That is, at decision epoch t_{k+1}, the Q-value $Q'(s_k, a_k)$ is updated according to the received reward as:

$$Q'(s_k, a_k) \leftarrow \underbrace{Q(s_k, a_k)}_{\text{old value}} + \underbrace{\beta_k}_{\text{learning rate}} \cdot \left[\overbrace{\underbrace{r_{k+1}}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_{a \in A} Q(s_{k+1}, a)}_{\text{max future value}}}^{\text{expected discounted reward}} - \overbrace{Q(s_k, a_k)}^{\text{old value}} \right], \quad (6)$$

where r_{k+1} is the expected reward at time t_{k+1} after taking action a_k at time t_k; $\gamma \in (0, 1)$ is the discount factor; and $\beta_k \in (0, 1)$ is the learning rate at time t_k. The next time state s is visited, the action with the maximum Q-value will be chosen, i.e., $\pi(s) = \max_{a \in A} Q(s, a)$, given that the Q-value was updated, it might lead to a different action than the one taken last time state s was visited. In this work, we set the discount factor as 0.28 and learning rate as 0.72. These factors are determined based on a wide range of experiments and set the values that yield the best performance.

5 RELIABILITY-AWARE POWER MANAGEMENT

In this section, we present the proposed Application- and Thermal-reliability-aware Power management by employing the previously discussed RL technique. One of the key challenges to perform power management considering the reliabilities is that application and thermal reliabilities have different units. For instance, time-dependent dielectric breakdowns are presented as parts per million (ppm) defective, whereas soft errors are quantified as failure in time (FIT) (Seifert et al. 2012; Swaminathan et al. 2017). As such, a direct combination of them is invalid.

As mentioned in the previous section, an RL agent performs near-optimal actions based on the current state and the corresponding immediate reward it gets. First, we define the state space, then the action space, followed by the way the reward is calculated in this work.

5.1 State Space

There exist various metrics, such as power or energy trace, memory access characteristics, priority of the application, CPU utilization rate, Cycles-Per-Instruction (CPI), and temperature, that serve as factors to perform multi-core power management and represent the current state of the system. As processing or employing all the metrics lead to computational overhead and can lead to convergence issues, a subset of them depending on the applied constraints is considered for power management. The power trace is a direct representation of the power/energy consumption and aids in performing efficient DVFS. As the state variables such as power consumption or reliability values are continuous in nature and can take any value, considering every value to represent a state might incur large computational complexity and hinder the convergence. To alleviate this, a set of discretized values is considered, and the original values are mapped to these discrete values of a state depending on how close the original value is to the discrete value. For instance, an original power consumption of 345mW will be mapped to a state having state value of 350mW. Here, the example is provided with just one variable in state, but in the simulations the state tuple has three values, as mentioned later. Furthermore, in contrast to other power management works, as this work also aims to meet the application- and thermal-reliability constraints, they are also considered to represent the state of the system here. It is non-trivial to consider these variables as the state of the system to ensure the overall reliability of the system.

Thus, the state of the system for the reinforcement learner (agent) are the per-application power trace and the corresponding reliability derived based on Equations (2) and (4). As such, each application has k states denoted by $s_1, s_2, \ldots, s_k$, where $s_1 < s_2 < \cdots < s_k$, i.e., arranged in terms of ascending order of power consumption. Each state here represents the power consumption of the running application and its reliability, i.e., $s_i = \{p_i, r_i, tr_i\}$, where power in the ith state is represented by p_i with corresponding application and thermal reliability as r_i, and tr_i, respectively.

5.2 Action Space

Each RL agent conducts a search into finite discrete space of possible target VF transitions as the action space, denoted by $A = \{a_1, a_2, \ldots, a_n\}$, where action a_i indicates assigning ith voltage and frequency levels (v_i, f_i) to the application. To avoid the convergence and complexity issues arising from the RL, we limit the number of feasible actions by having only four VF levels in this work.

5.3 Reward

The reward function has to be defined based on the state and the action taken by the RL agent. Thus, the reward has to be composed of the power consumption and reliability (thermal and application). As mentioned earlier, it is not straightforward to combine different reliabilities due to differences in their behaviors and cardinality. To overcome these concerns, works such as Swaminathan et al. (2017) proposed use of principal component analysis. Though effective, this is limited by a few factors such as non-linear or orthogonal relationship between application and thermal reliabilities, and the involved complexity to run in the utilized scenario. As such, we consider the variation in the reliabilities w.r.t. the desired reliability. The reward is calculated as a function of the reliability and energy savings. The reward associated with transitioning from state s to s' is given by

$$r_{k+1}|_{(s,a,s')} = \alpha_1(\Delta FR/FR_k) + \alpha_2(\Delta TR/TR_k) + \alpha_3(\Delta E/E_k), \qquad (7)$$

where s' indicates all the possible states from state s when action a is performed; $\Delta FR/FR_k$ and $\Delta TR/TR_k$ are the change in functional and thermal reliability w.r.t existing reliability when transitioned from state s to s' with action a; similarly, the difference in power consumption due to transition is given in second term ($\Delta E/E_k$). The α_1, α_2, and α_3 are the constants, set to 0.33 in this work. The functional and thermal reliability are derived based on Equations (2) and (3), respectively.

5.4 Power Management Policy Generation

We describe the power management policy generation by the RL agent here based on the described state, action, and the reward.

For an effective power management, the power management has to be proactive, as reactive power management is inefficient due to computational delays. We first predict the power trace based on the previous traces and generate the power management policy as follows: The input for the power management policy generator is the power trace of the system at application-level granularity. To facilitate a proactive runtime power management with less overhead, a linear predictor–based power trace prediction is performed first, as in Equation (8),

$$p(t+1) = \sum_{k=0}^{z} w_k p(t-k) + \epsilon,$$ (8)

where $p(t+1)$ represents the power at time-instant t, w_i represents the coefficient for regression, and the error is denoted by ϵ. In this work, the order is represented by z, set to 8 in experiments. The order is determined based on experiments to achieve lower error without overhead. With the chosen order, an average root mean square (RMSE) of 0.53 is achieved. Once the power is predicted, the corresponding reliability is derived, as given in Section 2.3. As the power trace is continuous in distribution, assigning each value to a state increases the computational complexity for the reinforcement learner. To avoid this computational complexity, the predicted power trace and the reliability is quantized and a state that has the closest power and reliability values to the fed predicted power and reliability is chosen as current state. The state is composed of power and reliability, i.e., state $s_i = \{p_i, r_i, tr_i\}$ where p_i denotes the power for state i, and corresponding reliabilities by r_i and tr_i, as described previously. As each application has k states denoted by $S = \{s_1, s_2, \ldots, s_k\}$, based on the predicted power and reliability, one of the states is assigned.

Based on the Bellman's principle of optimality (Bellman 2003), given the states, and reward function, the optimal policy can be derived as

$$\pi^*(s) = arg \max_a (Q'(s, a))$$ (9)

The $Q'(s, a)$ is presented in Equation (6). This $\pi^*(s)$ denotes the optimal policy for the system, given the system is in state s. As such, we generate the optimal state-action pairs based on the inputs. As the power management policy generation is performed offline and deployed online, the associated computational overhead does not impact power management. The proposed reliability-aware power management policy is not restricted to any specific type of reliability model or architecture and can be employed on different systems and with different reliability models.

An example of proposed Q-learning–based application- and thermal-reliability-aware power management is shown in the Figure 3. Based on the predicted power consumption, as given in Equation (8), and the derived application and thermal reliability for the given application, one of the states is mapped. For mapping, we consider the state with closest power consumption value. For instance, as shown in Figure 3, if the predicted power is 1.56W, then the closest state is s_4; as such, the current state is considered as state s_4. Further, depending on the current state and
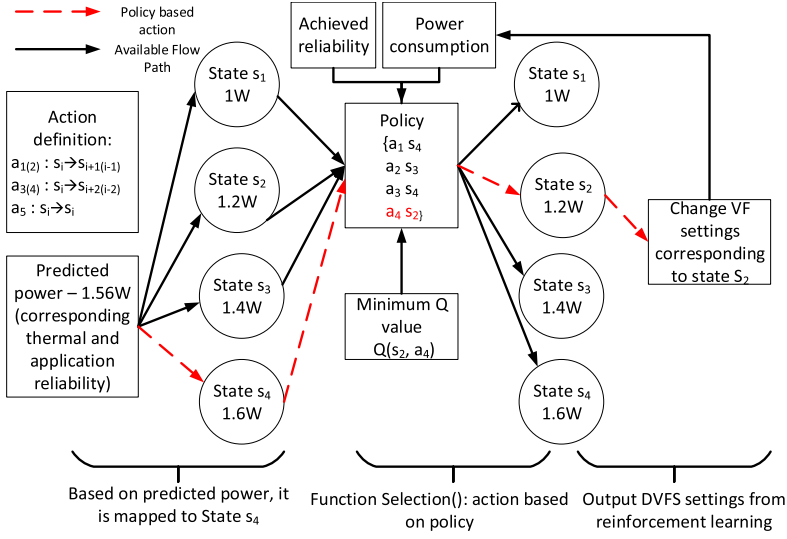
Fig. 3. An example describing the proposed application- and thermal-reliability-aware reinforcement learning–based power management.

the chosen policy based on Equation (9), one of the policies is chosen. The chosen policy and transitions are shown with a dotted line in Figure 3. Based on the chosen action and the power consumption and reliability variations, the new reward is calculated and fed to the policy maker. This process is repeated multiple times for convergence during the training phase. At the time of testing, as the policies are already pre-defined, the assignment happens in one iteration, leading to lower overhead. For the purpose of brevity, the reliabilities are not shown in Figure 3.

Summary

The whole process of RL-based application- and thermal-reliability-aware power management is outlined in Algorithm 1.

In the first step, based on the obtained power trace of an application, the power trace for future time-instants are predicted as in Line 1 of Algorithm 1. The corresponding reliability is derived for the application, as in Lines 2–3. Based on the predicted voltage and reliability, one of the states

---

ALGORITHM 1: Reliability-aware Power Management for multi-core system

---

Input: Power trace monitored at application-level granularity for all applications running (P), and runtime
Output: Voltage-Frequency (VF) settings
1: Predict power trace as $P(t+1) = \sum_{k=0}^{z} w_k P(t-k) + \epsilon$
2: Estimate corresponding application reliability, as in (2)
3: Estimate corresponding thermal reliability, as in (3)
4: Assign state for predicted power trace, and reliability, i.e., $\{p(t+1), r\} \rightarrow s_i, s_i \in S$
5: Calculate reward r_{k+1} as in (7)
6: Obtain the Q-values, as in (6)
7: Based on Bellman's principle, an action with optimal policy is derived as in (9)
8: The optimal policy provides the action to be taken, i.e., VF settings will be fed to DVFS controller for application-reliability-aware power management

---

Table 1. Overview of Core Configuration

Item	Description	Value
Microprocessor core	Frequency (Max)	2.0GHz
	Voltage (Max.)	1.0V
	Technology node	22nm
	L1-I cache	32KB
	L1-D cache	32KB
	L2 cache	256KB
L3-Cache	8MB	

are assigned, and the reward for the next time step based on all the possible actions for the given state is calculated and the corresponding Q-values are obtained, as given in Lines 4–6. Last, based on the Bellman's optimality principle, action with maximum Q-value is considered as optimal and fed to the DVFS controller to perform power management, as given in Lines 7–8 of Algorithm 1. In the simulations, we impose the constraints on the number of iterations performed for improved convergence.

6 SIMULATION RESULTS

Here, we present the simulation settings, followed by the experimental analysis and comparison with the existing traditional power management techniques.

6.1 System Settings

The proposed power management is implemented in Snipersim simulator (Carlson et al. 2014), which is a parallel, interval-accurate, high-speed, and accurate x86 simulator. Standard Intel Xeon microprocessor microarchitecture–based 22-nm core models are used in the simulations. The maximum voltage and frequency levels are 1.0V and 2.0GHz, respectively. In simulations, we use four voltage-frequency levels for power management, which are supported by standard Xeon processor microarchitecture–based cores: (1V, 2.0GHz), (0.9V, 1.8GHz), (0.8V, 1.5GHz), and (0.7V, 1.0GHz). However, this could be modified depending on the simulation environment and the utilized cores, and the proposed power management is independent of the underlying core architecture. To facilitate enough time for switching of VF levels and reduce the processing overhead of the monitored data, the application power traces are sampled at 10μs, though the time required for switching is in the range of few μs, as reported in Singhal (2008). Additional details on the configuration of microprocessor core and other components are presented in Table 1. To validate the power management, simulations are run with PARSEC (blackscholes, x264, bodytrack, swaptions, streamcluster, canneal, dedup, and fluidanimate applications are executed on the multi-core system) benchmark (Bienia et al. 2008). The number of cores is varied from 2 to 32 for simulations.

6.2 Performance Analysis

Here, we present the energy savings, runtime, and application reliability improvement with the proposed power manager and some other existing power management techniques.

6.2.1 Power Management at Different Abstraction Levels. The proposed technique focuses on power management at application-level. However, it is also possible to perform power management at lower abstraction level (core-level) and higher abstraction level (system-level or per-chip level). As a case study, we present the impact of power management at different abstraction levels for a four-core processor. For analysis, multi-threaded applications are chosen based on the manner
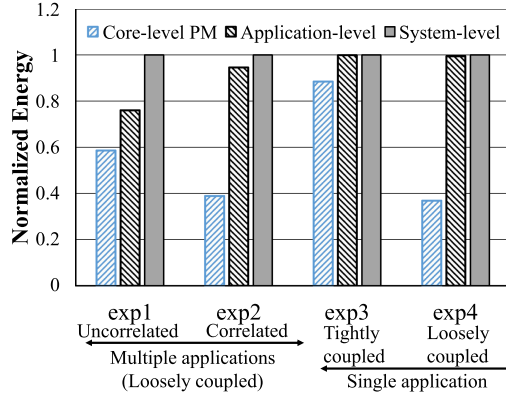
Fig. 4. Average power savings with proposed power management at different abstraction levels.

in which the workloads are distributed among cores. Two workload categories are chosen: (a) tightly coupled; and (b) loosely coupled workloads. Here, tightly coupled workload indicates that the workloads of an application are evenly distributed among multiple cores; and loosely coupled workload indicates that the workload of an application is unevenly distributed among multiple cores.

The normalized average power consumption at three different granularity levels for a microprocessor running multi-threaded application(s) is shown in Figure 4. Following are the observations:

- For loosely coupled multi-threaded applications, application level power management has better power savings compared to system level, if the applications are uncorrelated, i.e., applications are dissimilar.
- If the workloads are loosely coupled and correlated, i.e., similar workloads, then system-level and application-level power management achieve similar power savings.
- In case of single multi-threaded application (shown as single application in Figure 4) distributed among all the cores, irrespective of granularity, the power management achieves similar performance if the application is tightly coupled.
- For a loosely coupled application, system-level and application-level power management has similar performance.

As seen, per-core power management has better power savings; however, this adds additional overhead such as monitoring power regulators for each of the cores. System-level power management has lower overhead and reduced power savings compared to per-core power management. Per-application-level power management has performance in-between per-core and system-level power management. As running multiple applications that are dissimilar in nature is much realistic on multi-core systems, per-application–based power management is considered as a better choice for power management here. Some of the recent works have also shown that application-level is optimal for future multi-core power management and has lower overhead compared to per-core power management (Rahmani et al. 2017; Shafique et al. 2016), despite power saving with per-core being higher.

6.2.2 Energy Savings. To consider the power savings as well as performance (timing), we evaluate the effectiveness of proposed power management technique in terms of energy savings and compare the achieved energy savings of our proposed technique with other techniques. There are many prior techniques for power/energy management. We implemented a few, such as
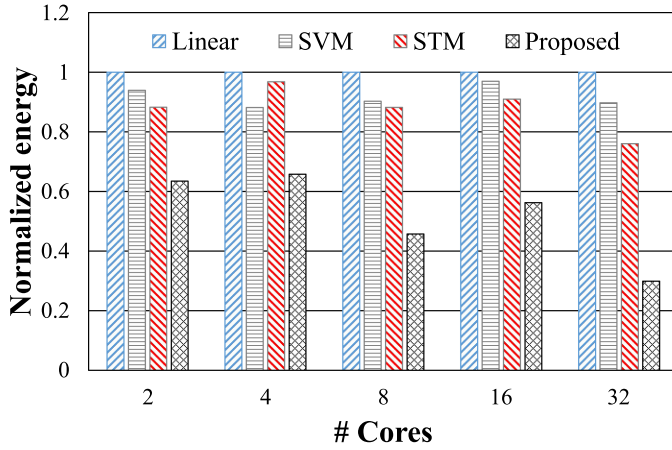
Fig. 5. Average energy consumption with proposed power management for microprocessor with different numbers of cores.

Manoj et al. (2015), Rountree et al. (2011), Yang et al. (2015), and Zaman et al. (2015) (with minor adaptations such as power management at application-level) for a fair comparison. The rationale for choosing these are as follows: In Manoj et al. (2015), prediction of workload using Auto-Regressive Moving Average (ARMA) and a Singular Value Decomposition (SVD)–based VF-level assignment is carried out, which has shown better scalability for future multi-core systems. Machine learning equipped power management is proposed in Zaman et al. (2015), where SVM-based regression for predicting workloads and SVM classifier–based VF-level assignment is employed. The sparse encoding is not implemented, as the data is not as large as that in the original work. A linear regression with offline learning or modeling-based workload prediction and VF-level assignment are utilized in Yang et al. (2015) and Rountree et al. (2011), which is lightweight in nature. Similar resemblances can be observed from other existing works.

Figure 5 presents the normalized energy consumption for multi-core system with 2, 4, 8, 16, and 32 cores. In Figure 5, X-axis represents the number of cores on which the benchmark applications are run and the Y-axis represents the normalized energy. In the legend of Figure 5, "Proposed," "Linear," "SVM," and "STM" represent the energy consumptions with proposed technique, linear regression–based power management (Rountree et al. 2011; Yang et al. 2015), SVM (Zaman et al. 2015), and space-time multiplexing (Manoj et al. 2015)-based power management techniques, respectively. For the experimental evaluation of proposed and other power management works, the benchmark applications are randomly assigned to cores.

The following observations can be made: For a system with a small number of cores (two cores), use of lightweight techniques (such as linear regression–based power management) is beneficial. However, for a large number of cores, proposed power manager has higher performance compared to other techniques. The rationale for these differences can be mentioned as follows:

- For miniature systems with two cores or less, the Q-learning adds higher computational overhead, i.e., the computations required to perform power management can incur more computations or overhead compared to execution of workloads without power management.
- For larger systems, the achieved energy savings are higher compared to the additional overhead.
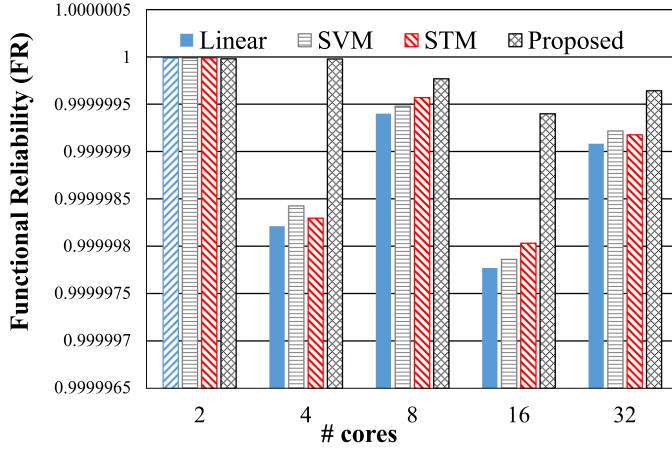
Fig. 6. Average application reliability with proposed power management and other power management works.

These observations clearly indicate that the proposed technique is scalable and beneficial for modern-day and future multi-core and many-core systems. On average, energy savings of 20% is achieved with our proposed technique compared to linear regression–based power management (Rountree et al. 2011; Yang et al. 2015) for a system with up to 32 cores. Similarly, an average energy savings of 11%, and 7.7% are achieved with our proposed power management technique compared to SVM (Zaman et al. 2015)-based and space-time multiplexing (Manoj et al. 2015)-based power management techniques.

6.3 Application Reliability

The employed reinforcement learning–based power manager not only considers power or energy savings as feedback (reward), but also considers the reliability of the application. Similar to energy savings, we compare the achieved application reliability with existing power management schemes.

Figure 6 presents the achieved application reliability with proposed RL-based power management and other power management works. One can observe that existing power-centric or performance-centric power management techniques have an impact on reliability as the energy savings improve.

- In contrast to the power-saving–oriented works, with the proposed power management, the reliability is also enhanced together with energy savings.

In this work, the Δ of Equation (1) is set to 1, and λ_0 is set to 10^{-6}, similar to Salehi et al. (2015). Even under optimal settings of having low functional vulnerability index ($FVI = 1$), the proposed RL-based power management achieves higher reliability compared to other prior techniques. In comparison with prior techniques that consider reliability for power management, the proposed technique has an advantage of learning the reliability variations with VF settings, and also learning characteristics makes proposed application- and thermal-reliability-aware power management achieve higher reliability. In comparison to linear regression, SVM, and STM-based power management, proposed power management has 1.8×, 1.99×, and 2.08× lower variance in terms of reliability, respectively, on average, for a microprocessor with up to 32 cores executing PARSEC applications. This is shown in Figure 6; lower variance indicates better stability.
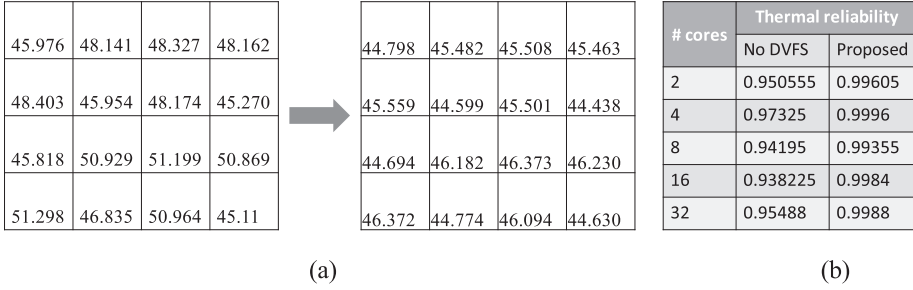
45.976	48.141	48.327	48.162
48.403	45.954	48.174	45.270
45.818	50.929	51.199	50.869
51.298	46.835	50.964	45.11

→

44.798	45.482	45.508	45.463
45.559	44.599	45.501	44.438
44.694	46.182	46.373	46.230
46.372	44.774	46.094	44.630

| # cores | Thermal reliability | |
	No DVFS	Proposed
2	0.950555	0.99605
4	0.97325	0.9996
8	0.94195	0.99355
16	0.938225	0.9984
32	0.95488	0.9988

(a)　　　　　　　　　　　　　　　　　　　　　　　　　　(b)

Fig. 7. (a) Reduction in temperature with the proposed power management; (b) improvement in thermal reliability of system.

6.4 Thermal Reliability

In addition to power savings and improvement in the application reliability, the proposed power management scheme as well considers the thermal reliability. This leads to improvement in the thermal reliability of the multi-core system. The thermal map at chip-level is obtained through McPAT tool. For the purpose of obtaining the thermal reliability at an application-level granularity, we consider the worst-case temperature for each application, i.e., for an application running on (say) cores 1, 2, and 4 with core 4 having maximum temperature among the three, we consider core 4's temperature for obtaining thermal reliability to account for worst-case scenario. The temperature reduction and thermal reliability improvements are shown in Figure 7. Figure 7(a) shows the thermal map of a 16-core processor. One can observe reduction in temperature with the proposed power management. A temperature reduction of up to 4.926 °C is observed. For the performed experiments with up to 32 cores, on average, a 2.193 °C reduction across cores is achieved. As most of the power management works are power-saving and application-reliability focused, for fairness, we did not compare the thermal savings with existing power management works. However, thermal management works are temperature-focused rather than power-saving-focused, hence a comparison will be unfair.

In addition to reduction temperature, improvement in thermal reliability is also observed, as shown in Figure 7(b). On average, 99.73% thermal reliability is achieved with the proposed power management, which is nearly 5% higher, on average, compared to the multi-core system without any power management. Though the numbers might look small in terms of difference, this difference can become higher when the system is run for longer periods of time, due to accumulated heat.

- Thus, in addition to the energy savings and reliability enhancement, the proposed power management scheme can also result in lower on-chip temperatures, leading to higher efficiency.

6.5 Overhead Analysis

As the proposed power management technique involves switching and computations (needed to predict the VF levels), it adds overheads to the system, which we discuss here. We measure the execution time of the application without any power management technique and under different power management techniques. The additional execution time can be considered as the overhead caused due to involved computations and VF switching. The average runtime for all the executed benchmark applications on multi-core systems with 2 to 32 cores under different power management techniques is outlined in Table 2, obtained from McPAT of SniperSim. Compared to a system that has no power management, proposed power management adds nearly 24% overhead in terms

Table 2. Average Runtime (in Seconds) for
Applications Running on Multi-core System

No DVFS	Linear	SVM	STM	Proposed
0.101	0.149	0.118	0.130	0.124

of runtime. However, compared to power management techniques such as linear regression, SVM, and STM, our proposed technique has 22.3%, −6%, and 5.4% reduced runtime, respectively. In the experiments, the linear regression–based power management has to be performed with a large order to achieve similar power savings, leading to larger runtime. The reduced runtime with our proposed technique is because of embedded learning in the proposed power management of application characteristics and reliability. We anticipate that the runtime for SVM is lower than our proposed technique due to the involved complexity.

7 CONCLUSION

Existing power management techniques perform power management under the constraints of power or performance budgets. However, application reliability is impacted by lowering voltage frequency, and thermal reliability is exacerbated with increase in voltage-frequency levels. In response, we proposed an application- and thermal-reliability-aware reinforcement learning–based multi-core power management technique. In the proposed power management technique, the power trace monitored at application-level granularity is fed to the reinforcement learner (Q-learner) along with the application and thermal reliability. The Q-learner optimizes the VF settings for the next time period for the application, considering both reliability and power consumption (defined in reward function). With the proposed technique, an energy savings of up to 20% on average, no degradation in application reliability (up to 2.08× lower variation in application reliability), up to 4.926 °C temperature reduction, and lower runtime is achieved when compared with existing power management techniques.

REFERENCES

A. Bartolini, M. Cacciari, A. Tilli, and L. Benini. 2013. Thermal and energy management of high-performance multi-cores: Distributed and self-calibrating model-predictive controller. *IEEE Trans. Parallel Distrib. Syst.* 24, 1 (Jan. 2013), 170–183.

A. Bartolini et al. 2011. A distributed and self-calibrating model-predictive controller for energy and thermal management of high-performance multi-cores. In *Proceedings of the Design, Automation and Test in Europe Conference (DATE'11)*.

Richard Ernest Bellman. 2003. *Dynamic Programming.* Dover Publications, Incorporated.

R. Bergamaschi et al. 2008. Exploring power management in multi-core systems. In *Proceedings of the Asia and South Pacific Design Automation Conference*.

Christian Bienia et al. 2008. The PARSEC Benchmark suite: Characterization and architectural implications. In *Proceedings of the International Conference on Parallel Architectures and Compilation Techniques*.

D. Brooks, R. P. Dick, R. Joseph, and L. Shang. 2007. Power, thermal, and reliability modeling in nanometer-scale microprocessors. *IEEE Micro* 27, 3 (May 2007), 49–62.

Trevor E. Carlson et al. 2014. An evaluation of high-level mechanistic core models. *ACM Trans. Archit. Code Optim.* 11, 3 (Aug. 2014), 28:1–28:25.

Kihwan Choi, Ramakrishna Soma, and Massoud Pedram. 2004. Dynamic voltage and frequency scaling based on workload decomposition. In *Proceedings of the International Symposium on Low Power Electronics and Design*.

Kihwan Choi, R. Soma, and M. Pedram. 2005. Fine-grained dynamic voltage and frequency scaling for precise energy and performance tradeoff based on the ratio of off-chip access to on-chip computation times. *IEEE Trans. Comput.-Aided Des. Integr. Circ. Syst.* 24, 1 (Jan. 2005), 18–28.

Foad Dabiri, Ani Nahapetian, Miodrag Potkonjak, and Majid Sarrafzadeh. 2007. Soft error-aware power optimization using gate sizing. In *Integrated Circuit and System Design: Power and Timing Modeling, Optimization and Simulation (PATMOS'07)*, N. Azémard and L. Svensson (Eds.). Lecture Notes in Computer Science, Vol. 4644. Springer, Berlin, Heidelberg.

B. Dietrich et al. 2010. LMS-based low-complexity game workload prediction for DVFS. In *Proceedings of the IEEE International Conference on Computer Design*.

A. Ejlali, B. M. Al-Hashimi, and P. Eles. 2012. Low-energy standby-sparing for hard real-time systems. *IEEE Trans. Comput.-Aided Des. Integr. Circ. Syst.* 31, 3 (Mar. 2012), 329–342.

Hadi Esmaeilzadeh et al. 2011. Dark silicon and the end of multicore scaling. In *Proceedings of the International Symposium on Computer Architecture*.

D. Gnad, M. Shafique, F. Kriebel, S. Rehman, and J. Henkel. 2015. Hayat: Harnessing dark silicon and variability for aging deceleration and balancing. In *Proceedings of the Design Automation Conference (DAC'15)*.

Wei Huang, Shougata Ghosh, Siva Velusamy, Karthik Sankaranarayanan, Kevin Skadron, and Mircea R. Stan. 2006. Hotspot: Acompact thermal modeling methodology for early-stage VLSI design. *IEEE Trans. Very Large Scale Integr. Syst.* 14, 5 (May 2006), 501–513.

H. Jung and M. Pedram. 2010. Supervised learning based power management for multicore processors. *IEEE Trans. Comput.-Aided Des. Integr. Circ. Syst.* 29, 9 (Sept. 2010), 1395–1408. DOI : https://doi.org/10.1109/TCAD.2010.2059270

N. Kapadia and S. Pasricha. 2015. VARSHA: Variation and reliability-aware application scheduling with adaptive parallelism in the dark-silicon era. In *Proceedings of the Design, Automation Test in Europe Conference Exhibition (DATE'15)*.

J. S. Lee, K. Skadron, and S. W. Chung. 2010. Predictive temperature-aware DVFS. *IEEE Trans. Comput.* 59, 1 (Jan. 2010), 127–133.

S. Li, J. H. Ahn, R. D. Strong, J. B. Brockman, D. M. Tullsen, and N. P. Jouppi. 2009. McPAT: An integrated power, area, and timing modeling framework for multicore and manycore architectures. In *Proceedings of the IEEE/ACM International Symposium on Microarchitecture (MICRO'09)*.

W. Liu, Y. Tan, and Q. Qiu. 2010. Enhanced Q-learning algorithm for dynamic power management with performance constraint. In *Proceedings of the Design, Automation and Test in Europe Conference (DATE'10)*. 602–605. DOI : https://doi.org/10.1109/DATE.2010.5457135

Shiting (Justin) Lu, Russell Tessier, and Wayne Burleson. 2015. Reinforcement learning for thermal-aware many-core task allocation. In *Proceedings of the Great Lakes Symposium on VLSI*.

M. A. Makhzan, A. Khajeh, A. Eltawil, and F. Kurdahi. 2007. Limits on voltage scaling for caches utilizing fault tolerant techniques. In *Proceedings of the International Conference on Computer Design*.

P. D. Sai Manoj, A. Jantsch, and M. Shafique. 2018. SmartDPM: Dynamic power management using machine learning for multi-core microprocessors. *J. Low-Power Electron.* 14, 4 (Dec. 2018).

P. D. Sai Manoj, J. Lin, S. Zhu, Y. Yin, X. Liu, X. Huang, C. Song, W. Zhang, M. Yan, Z. Yu, and H. Yu. 2017. A scalable network-on-chip microprocessor with 2.5D integrated memory and accelerator. *IEEE Trans. Circ. Syst. I: Reg. Papers* 64, 6 (June 2017), 1432–1443.

P. D. Sai Manoj, H. Yu, H. Huang, and D. Xu. 2016. A Q-Learning based self-adaptive I/O communication for 2.5D integrated many-core microprocessor and memory. *IEEE Trans. Comput.* 65, 4 (Apr. 2016), 1185–1196.

P. D. Sai Manoj, H. Yu, Y. Shang, C. S. Tan, and S. K. Lim. 2013. Reliable 3-D clock-tree synthesis considering nonlinear capacitive TSV model with electrical-thermal-mechanical coupling. *IEEE Trans. Comput.-Aided Des. Integr. Circ. Syst.* 32, 11 (Nov. 2013), 1734–1747.

P. D. Sai Manoj, H. Yu, and K. Wang. 2015. 3D Many-core microprocessor power management by space-time multiplexing based demand-supply matching. *IEEE Trans. Comput.* 64, 11 (Nov. 2015), 3022–3036.

S. S. Mukherjee, M. Kontz, and S. K. Reinhardt. 2002. Detailed design and evaluation of redundant multi-threading alternatives. In *Proceedings of the International Symposium on Computer Architecture*.

S. Pagani et al. 2017. Energy efficiency for clustered heterogeneous multicores. *IEEE Trans. Parallel Distrib. Syst.* 28, 5 (May 2017), 1315–1330.

S. Pagani, H. Khdr, W. Munawar, J. Chen, M. Shafique, M. Li, and J. Henkel. 2014. TSP: Thermal safe power—Efficient power budgeting for many-core systems in dark silicon. In *Proceedings of the International Conference on Hardware/Software Codesign and System Synthesis*.

S. Pagani, P. D. Sai Manoj, A. Jantsch, and J. Henkel. 2018. Machine learning for power, energy, and thermal management on multi-core processors: A survey. *IEEE Trans. Comput.-Aided Des. Integ. Circ. Syst.* PP, 1–17. DOI : 10.1109/TCAD.2018.2878168

X. Qi, D. Zhu, and H. Aydin. 2010. Global reliability-aware power management for multiprocessor real-time systems. In *Proceedings of the IEEE International Conference on Embedded and Real-Time Computing Systems and Applications*.

Amir M. Rahmani et al. 2017. Reliability-aware runtime power management for many-core systems in the dark silicon era. *IEEE Trans. Very Large Scale Integr. Syst.* 25, 2 (Feb. 2017), 427–440.

Krishna K. Rangan, Gu-Yeon Wei, and David Brooks. 2009. Thread motion: Fine-grained power management for multi-core systems. *SIGARCH Comput. Archit. News* 37, 3 (Jun. 2009), 302–313.

B. Rountree et al. 2011. Practical performance prediction under dynamic voltage frequency scaling. In *Proceedings of the International Green Computing Conference and Workshops*.

M. Salehi et al. 2015. dsReliM: Power-constrained reliability management in dark-silicon many-core chips under process variations. In *Proceedings of the International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS'15)*.

M. Salehi, M. K. Tavana, S. Rehman, F. Kriebel, M. Shafique, A. Ejlali, and J. Henkel. 2015. DRVS: Power-efficient reliability management through dynamic redundancy and voltage scaling under variations. In *Proceedings of the International Symposium on Low Power Electronics and Design*.

Avesta Sasan, Houman Homayoun, Ahmed Eltawil, and Fadi Kurdahi. 2009. A fault tolerant cache architecture for sub 500mV operation: Resizable data composer cache (RDC-cache). In *Proceedings of the International Conference on Compilers, Architecture, and Synthesis for Embedded Systems*.

N. Seifert, B. Gill, S. Jahinuzzaman, J. Basile, V. Ambrose, Q. Shi, R. Allmon, and A. Bramnik. 2012. Soft error susceptibilities of 22 nm tri-gate devices. *IEEE Trans. Nucl. Sci.* 59, 6 (Dec. 2012), 2666–2673.

Muhammad Shafique, Siddharth Garg, Jörg Henkel, and Diana Marculescu. 2014. The EDA challenges in the dark silicon era: Temperature, reliability, and variability perspectives. In *Proceedings of the Design Automation Conference*.

M. Shafique, A. Ivanov, B. Vogel, and J. Henkel. 2016. Scalable power management for on-chip systems with malleable applications. *IEEE Trans. Comput.* 65, 11 (Nov. 2016), 3398–3412.

H. Shen, J. Lu, and Q. Qiu. 2012. Learning-based DVFS for simultaneous temperature, performance and energy management. In *Proceedings of the International Symposium on Quality Electronic Design (ISQED'12)*.

Hao Shen, Ying Tan, Jun Lu, Qing Wu, and Qinru Qiu. 2013. Achieving autonomous power management using reinforcement learning. *ACM Trans. Des. Auto. Electron. Syst.* 18, 2 (Apr. 2013), 24:1–24:32. DOI : https://doi.org/10.1145/2442087.2442095

A. Shye, T. Moseley, V. J. Reddi, J. Blomstedt, and D. A. Connors. 2007. Using process-level redundancy to exploit multiple cores for transient fault tolerance. In *Proceedings of the IEEE/IFIP International Conference on Dependable Systems and Networks*.

R. Singhal. 2008. Inside Intel® core microarchitecture (Nehalem). In *Proceedings of the IEEE Hot Chips Symposium*.

J. Srinivasan, S. V. Adve, P. Bose, and J. A. Rivers. 2004. The impact of technology scaling on lifetime reliability. In *Proceedings of the International Conference on Dependable Systems and Networks*.

Jayanth Srinivasan, S. V. Adve, Pradip Bose, and J. A. Rivers. 2005. Lifetime reliability: Toward an architectural solution. *IEEE Micro* 25, 3 (May 2005), 70–80.

K. Swaminathan, N. Chandramoorthy, C. Y. Cher, R. Bertran, A. Buyuktosunoglu, and P. Bose. 2017. BRAVO: Balanced reliability-aware voltage optimization. In *Proceedings of the IEEE International Symposium on High Performance Computer Architecture (HPCA'17)*.

Ying Tan, Wei Liu, and Qinru Qiu. 2009. Adaptive power management using reinforcement learning. In *Proceedings of the International Conference on Computer-Aided Design (ICCAD'09)*. 461–467. DOI : https://doi.org/10.1145/1687399.1687486

S. J. Tarsa, A. P. Kumar, and H. T. Kung. 2014. Workload prediction for adaptive power scaling using deep learning. In *Proceedings of the IEEE International Conference on IC Design Technology*.

Yanzhi Wang et al. 2011. Deriving a near-optimal power management policy using model-free reinforcement learning and Bayesian classification. In *Proceedings of the 48th Design Automation Conference (DAC'11)*.

Y. Wang and M. Pedram. 2016. Model-free reinforcement learning and Bayesian classification in system-level power management. *IEEE Trans. Comput.* 65, 12 (Mar. 2016), 3713–3726.

E. Wu, J. Suñé, W. Lai, E. Nowak, J. McKenna, A. Vayshenker, and D. Harmon. 2002. Interplay of voltage and temperature acceleration of oxide breakdown for ultra-thin gate oxides. *Solid-State Electron.* 46, 11 (2002), 1787–1798.

K. Wu and D. Marculescu. 2014. Power-planning-aware soft error hardening via selective voltage assignment. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* 22, 1 (Jan. 2014), 136–145.

S. S. Wu, K. Wang, P. D. Sai Manoj, T. Y. Ho, M. Yu, and H. Yu. 2014. A thermal resilient integration of many-core microprocessors and main memory by 2.5D TSI I/Os. In *Proceedings of the Design, Automation Test in Europe Conference Exhibition (DATE'14)*.

D. Xu, N. Yu, P. D. Sai Manoj, K. Wang, H. Yu, and M. Yu. 2015. A 2.5-D Memory-logic integration with data-pattern-aware memory controller. *IEEE Design Test* 32, 4 (Aug. 2015), 1–10.

X. Xu, K. Teramoto, A. Morales, and H. H. Huang. 2013. DUAL: Reliability-aware power management in data centers. In *Proceedings of the IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing*.

Sheng Yang et al. 2015. Adaptive energy minimization of embedded heterogeneous systems using regression-based learning. In *Proceedings of the International Workshop on Power and Timing Modeling, Optimization and Simulation*.

M. Zaman et al. 2015. Workload characterization and prediction: A pathway to reliable multi-core systems. In *Proceedings of the IEEE International On-Line Testing Symposium*.

Dakai Zhu, R. Melhem, and D. Mosse. 2004. The effects of energy management on reliability in real-time embedded systems. In *Proceedings of the IEEE/ACM International Conference on Computer Aided Design*.