Dynamic Thermal Management with Proactive Fan Speed Control Through Reinforcement Learning

Arman Iranfar*, Federico Terraneo†, Gabor Csordas*, Marina Zapater*, William Fornaciari†, David Atienza*

*Embedded Systems Laboratory (ESL), Swiss Federal Institute of Technology Lausanne (EPFL), Switzerland
{arman.iranfar, gabor.csordas, marina.zapater, david.atienza}@epfl.ch
†Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Italy
{federico.terraneo, william.fornaciari}@polimi.it

***Abstract*—Dynamic Thermal Management (DTM) has become a major challenge since it directly affects Multiprocessors Systems-on-chip (MPSoCs) performance, power consumption, and reliability. In this work, we propose a transient fan model, enabling adaptive fan speed control simulation for efficient DTM. Our model is validated through a thermal test chip achieving less than $2^{\circ}C$ error in the worst case. With multiple fan speeds, however, the DTM design space grows significantly, which can ultimately make conventional solutions impractical. We address this challenge through a reinforcement learning-based solution to proactively determine the number of active cores, operating frequency, and fan speed. The proposed solution is able to reduce fan power by up to 40% compared to a DTM with constant fan speed with less than 1% performance degradation. Also, compared to a state-of-the-art DTM technique our solution improves the performance by up to 19% for the same fan power.**

I. Introduction

With the advent of submicron technology, temperature has become one of the major challenges for Multiprocessor Systems-on-Chip (MPSoCs). Hot spots, thermal cycles, as well as temporal and spatial thermal gradients can significantly affect MPSoCs lifetime reliability, power consumption, performance, and cooling cost [1], [2]. Although all modern processors take advantage of built-in thermal protection to shut down the processor if a thermal emergency occurs [3], mechanism are needed to optimize other design objectives, such as performance. As a result, Dynamic Thermal Management (DTM) techniques have been widely applied [4] through different approaches, such as Dynamic Voltage and Frequency Scaling (DVFS) [5], task migration and allocation. In addition to such DTM techniques, variable fan speed contributes to hot spot alleviation [6], and provides a knob to control temperature without reducing performance. Although in most platforms fan speed can be dynamically adjusted for more efficient heat removal, many DTM approaches do not consider it [2], or simply rely on the default settings handled by the OS [7], which could be sub-optimal since it does not simultaneously consider all available runtime parameters. Moreover, once fan speed is considered, it is vital to take into account its power consumption as it may account for more than 20% of the power consumption in a typical server [8].

This work has been supported by the EC H2020 RECIPE project (GA No. 801137), and the ERC Consolidator Grant COMPUSAPIEN (GA No. 725657).

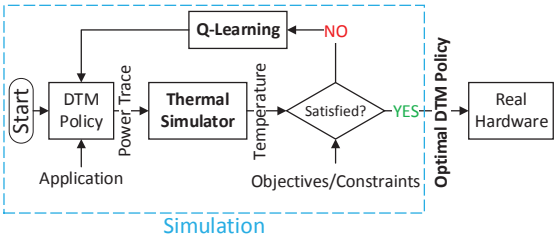

Fig. 1. Simulation framework and methodology for learning process

One of the main reasons that makes researchers reluctant to consider fan speed as a DTM knob is the lack of fan support in MPSoC transient thermal simulators. In this work, we integrate a fan model into the open-source 3D-ICE simulator [9] which enables simulating fan speed effects on the chip temperature.

With different fan speeds added to the DTM knobs, the number of runtime design parameters to be decided dynamically increases, possibly leading to challenges in find the best values to optimize the system behavior. As a result of such explosive design space, conventional solutions such as grid search, and offline look-up tables [10], are infeasible, impractical, or insufficient. On the contrary, machine learning has recently proved to be promising in efficiently searching large design spaces. Therefore, in this work, we adapt reinforcement learning (RL) and, in particular, Q-Learning (QL) to effectively control the processor temperature by selecting DVFS points, fan speed and the number of active cores. In particular, we let the QL agent explore the design space within the open-source 3D-ICE thermal simulation framework equipped with fan speed control, and learn the optimal DTM policy. Afterwards, we apply the learned DTM policy online on a real thermal test chip [11], proving the feasibility of our approach.

The main contributions of this paper are as follows:

- We integrate an accurate fan model into one of the most well-known open-source thermal simulators, 3D-ICE, which allows to assess DTM control policies.
- We validate our transient fan model on a thermal test chip under different power mappings with the maximum mean absolute error of less than $2^{\circ}C$ in the worst case.
- We propose a reinforcement learning-based DTM policy that leverages fan speed, DVFS and dynamic core assignment which jointly maximizes the performance and minimizes fan power subject to thermal constraints.

- We validate our proposed RL-based DTM policy on the thermal test chip.
- Our results show that the proposed proactive fan speed control can save up to 40% of cooling power, with less than 1% performance degradation in comparison to a DTM policy with a fixed fan speed. Also, we show that our RL-based solution can achieve up to 19% performance improvement, with the same average fan power and no thermal constraint violations when compared to state of the art techniques.

II. Related Work

A. Thermal Simulators and Analytic Models

There are various chip thermal simulators in the literature, each with unique features. However, none of them properly supports forced convection for transient simulation with variable fan speed such that it can be used to explore fan-based DTM policies. For instance, Therminator [12] is a thermal simulator for smartphones that only considers steady state. Manchester Thermal Analyzer (MTA) [13] supports transient simulations, however, it does not support the simulation of fans, but only heat sinks. HotSpot [14] is one the most well-known thermal simulators. However, it uses a single lumped convection resistance to model a simple heat sink, air flow, and fan which is inadequate if fan speed is to be changed during simulation (as in DTM). Finally, 3D-ICE [9] is a recent open-source thermal simulator that enables evaluation of advanced configurations, such as 3D chips with microchannel cooling, but it uses only a simple model for the heat sink. Unfortunately, none of these thermal simulators in their current form can be used as a comprehensive framework for DTM since fan speed control has been disregarded.

A few other works have modelled fan speed impact on temperature; for instance, [15] considers a variable thermal resistance to model heat sink and fan. However, their model does not provide any transient analysis and is limited to steady state. Finally, although a few works exist proposing transient thermal models of fans [8], they are coarse grain in nature, for example modeling the CPU and heat sink as uniform in temperature and are thus more suitable for high-level thermal modeling in data centers, rather than for DTM exploration.

B. DTM with Adaptive Fan Control

A large number Dynamic Thermal Management (DTM) policies have been proposed in the literature [16]. However, many of them discard active cooling [17], or simply consider its power consumption in a power model [18], rather than leveraging adaptive fan speed control schemes to further improve DTM efficiency.

Nevertheless, a few works consider fan speed as a knob. In particular, an optimization framework is proposed in [19] to find the optimal fan speed. Nonetheless, the framework has not been validated through real measurements and its applicability when the number of cores scales remains in question. TECfan [20] and [10] uses a look-up table created offline for different fan speeds and workloads. However, the provided DTM policy of such an approach is limited to the studied workloads and may not be applied to unseen workloads. Authors in [21] formulate the multiprocessor temperature as a convex function of fan speed and solve three optimization problems separately in different time scales to find the optimal task migration, DVFS, and fan speed. In this solution, however, the number of released tasks determines the active cores. Thus, the number of cores and performance could be non-optimal. Core temperatures are estimated using neural networks for preemptive fan control in [22]. In this work, authors consider very large intervals (in order of minutes) for workload variations such that each core temperature reaches its steady state. Unfortunately, for many applications workload may vary in order of seconds or even less [23]. Authors in [24] define a convex optimization problem to find the optimal fan speed. This work, however, does not take into account other system settings, such as number of cores and core frequency. Finally, Kim et al. [25] propose a fan speed controller to guarantee server operation stability without directly considering optimization of fan power.

III. Forced Convection Model

Performing closed-loop thermal simulations where the DTM policy can control the fan speed requires a thermal model capable of accurately modeling not only the chip, but also the heat sink and convective heat transfer to the air.

Our thermal model uses a finite volume approach to model thermal conduction in the heat sink base and fins, and a computationally efficient forced convection model. The heat sink is divided in individual volumes which are simulated using the following equation

$$C\frac{dT}{dt} = \sum_{i=1}^{n} G_{solid,i}(T_i - T) + \sum_{j=1}^{m} G_{air,j}(t)(T_{air} - T). \quad (1)$$

In this equation, T is the volume temperature, t is time, C, $G_{solid,i}$ and $G_{air,j}(t)$ are respectively the volume thermal capacity and conductance towards neighboring volumes and air, while T_i and T_{air} are the temperature of neighboring volumes and air. Parameters n, the number of faces of the volume that are adjacent to other solid volumes, and m, the number of faces of the volume open to airflow are set on a per volume basis depending on its location in the heat sink geometry. Obviously, $n + m \leq 6$, and if $n + m < 6$ the remaining faces are assumed adiabatic.

Of the coefficients of equation (1), all but $G_{air,j}(t)$ can be computed offline based on the heat sink geometry and material properties, while $G_{air,j}(t)$ is computed as:

$$G_{air,j} = \alpha V(t)^{\beta} \quad (2)$$

where $V(t)$ is the air velocity as a function of time and α and β are parameters that are computed offline based on the fin geometry, spacing, fan speed range and consequently expectation for the air flow to be laminar, turbulent or transition between the two regimes along the fin length.

This model introduces a major challenge, namely, the dependence of the thermal conductance on the fan velocity is nonlinear and cannot be reasonably approximated as a linear function over a sufficiently wide range of air velocities to

support DTM with variable fan speeds. However, existing chip thermal simulators only support linear equations.

To overcome this challenge we introduce a new parametric heat sink model accounting for forced convection. This model is written in Modelica [26], a domain specific language for solving differential equations. The use of Modelica increases the abstraction level at which a thermal model can be written, as it allows to write the differential equations directly, as opposed to writing the code to numerically solve them. A Modelica compiler uses symbolic manipulation to transform the equations in C code, automating the complex and error-prone task of manually writing optimized code to solve differential equations. To perform the Modelica to C translation we use the open source OpenModelica [27] compiler.

The heat sink model is integrated with the existing 3D-ICE thermal simulator, whose purpose is to simulate the chip active silicon, bulk, and heat spreader. The heat sink model and 3D-ICE operate in concert, at each simulated time step the heat sink model and 3D-ICE model exchange the temperature of the adjacent volumes and heat fluxes, in order for the two simulators to operate as one. This approach allows to leverage both the highly optimized code of 3D-ICE for simulating the linear part of the system (the silicon), and the nonlinear-capable Modelica code for the heatsink model. These co-simulation approaches are frequently used in MPSoC simulators, for instance, to optimize the DVFS management strategies [5].

IV. Thermal Management of Multicore Processors with Proactive Fan Speed Control

A. Proposed Reinforcement Learning-Based Approach

Reinforcement Learning is a machine learning approach that deals with environment-dependent problems through dynamic optimization programming [28]. In particular, RL employs an agent capable of taking actions from a finite action set, **A**, observing states from a state space, **S**, and using a reward as a result of selecting an action in a particular state to further modify its future behavior in the environment. In this work, we leverage Q-Learning (QL), a model-free algorithm of reinforcement learning. In QL, the agent learns the best policy, π, to apply an action in each specific state by storing a Q-value for each state-action pair as $Q^{\pi}(s,a)$. Q-values are updated after a new state and the obtained reward are observed, as follows:

$$Q_{t+1}(s_t,a_t) = [1-\alpha] \times Q_t(s_t,a_t) + \\ \alpha \times [R_{t+1} + \gamma \max_a Q_t(s_{t+1},a)] \quad (3)$$

where $Q_t(s_t,a_t)$ and $Q_{t+1}(s_t,a_t)$ are, respectively, the current and updated Q-values corresponding to the current taken action a_t and at the current state s_t, R_{t+1} is the immediate reward after next state s_{t+1} is observed, α determines the learning rate, and γ is the discount factor.

Although RL-based solutions are not rare in MPSoCs power and thermal management [29], they have not been used to incorporate fan speed control. In what follows, we explain, in detail, different components of our QL-based solution, i.e., actions, states, reward, and learning process.

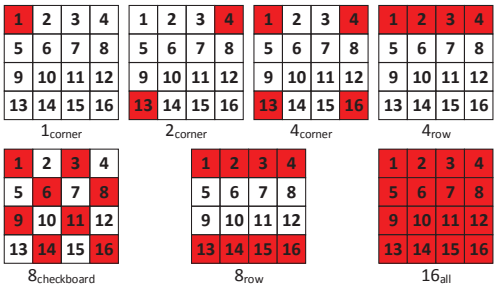

Fig. 2. Number of cores and corresponding mapping

1) Actions: The available actions to the agent consist of number of cores (N_c), operating frequency (F_c), and fan speed (S). In particular, at each decision point, the agent specifies a tuple of (N_c, F_c, S) with $N_c \in \{1_{corner}, 2_{corner}, 4_{corner}, 4_{row}, 8_{row}, 8_{checkerboard}, 16_{all}\}$, $F_c \in \{F_{min}, F_{mid}, F_{max}\}$, and $S \in \{S_{min}, S_{mid}, S_{max}\}$. The subscripts in N_c represent the corresponding application mapping on the multicore platform, as shown in Fig. 2. Note that the available actions to the RL agent are not limited to those suggested, and can be any arbitrary parameter.

2) States: We define the states based on the current hot-spot temperature ($\theta_h(t)$) and its gradient in the last time interval defined as $\frac{\Delta\theta_h}{\Delta t}$. We consider two constant coefficient, $\beta_1 < \beta_2 < 1$, to specify how close the current temperature is to the critical one. Finally, we define 6 different states as follows, where in the first four, $\theta_h < \beta_2\theta_{crit}$:

State 1: $\theta_h(t) > \beta_1\theta_{crit}$ and $\frac{\Delta\theta_h}{\Delta t} > 0$
State 2: $\theta_h(t) > \beta_1\theta_{crit}$ and $\frac{\Delta\theta_h}{\Delta t} < 0$
State 3: $\theta_h(t) < \beta_1\theta_{crit}$ and $\frac{\Delta\theta_h}{\Delta t} > 0$
State 4: $\theta_h(t) < \beta_1\theta_{crit}$ and $\frac{\Delta\theta_h}{\Delta t} < 0$
State 5: $\theta_h(t) > \beta_2\theta_{crit}$ and $\frac{\Delta\theta_h}{\Delta t} < 0$
State 6: $\theta_h(t) > \beta_2\theta_{crit}$ and $\frac{\Delta\theta_h}{\Delta t} > 0$

3) Reward Function: Since our goal is to maximize the performance and minimize fan power under the thermal constraint, we propose the following reward function:

$$R = \begin{cases} -2 & \theta > \theta_{crit} \\ \zeta_1 \frac{perf}{perf_{max}} + (1-\zeta_2 \frac{P_{fan}}{P_{fan,max}}) & otherwise \end{cases} \quad (4)$$

where $0 < \zeta_i < 1$ is a constant denoting the significance of each objective set by the designer, and $\Sigma\zeta_i = 1$.

4) Learning Process: In this work, we consider an $\epsilon - greedy$ policy where the agent takes a random action with probability ϵ and selects an action from already taken ones with probability $1-\epsilon$. Since, in the beginning, the agent starts from the *exploration* phase and no action has been chosen, ϵ is equal to 1. In order to let the agent gradually move from the pure exploration to the exploitation phase, we follow the schedule shown in TABLE I.

B. Simulation Framework and methodology

We use a simulation framework to accomplish the learning process as shown in Fig. 3. At each decision point, the RL

TABLE I
SCHEDULE OF ϵ BASED ON NUMBER OF ACTIONS TO BE TAKEN

ϵ	1	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1
# **action**	70	60	50	40	35	30	25	20	15	10

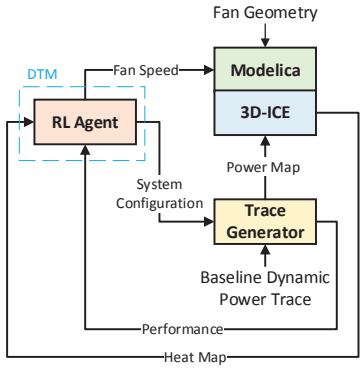

Fig. 3. Simulation framework and methodology for learning process

agent selects a fan speed and system configuration, namely, number of active cores and core frequency. The fan speed along with the fan geometry are used by the Modelica library described in Section III to simulate heat transfer through the heat sink. 3D-ICE advances the chip thermal simulation with respect to power traces corresponding to the configuration system and the baseline dynamic power trace (i.e., baseline workload). To generate these traces from a baseline workload, we use a simplified power and performance model described in V-C. Once the new decision time arrives, the RL agent receives the 3D-ICE output (heat map), application performance, and fan power, such that, new state is determined and the reward corresponding to the previous state and taken action is calculated. Such a framework lets the agent explore various system configurations under different conditions (workload, initial temperature, etc.) and optimize its behavior in the environment. Once the optimal behavior is learned, it can be used online on a real chip, as a DTM policy.

V. EXPERIMENTAL SETUPS

A. Thermal Test Chip

Validation of transient thermal models is a complex problem due to the need to apply known spatial and temporal power profiles to a chip connected to the desired heat dissipation solution, as well as to measure the resulting temperature maps of the active silicon layer. Ordinary MPSoCs and multicore processors are unsuitable for this task, as they generally have few temperature sensors, with significant noise and their exact location on the silicon die is not publicly known. Moreover, it is not possible to apply a prescribed power spatial distribution to off-the-shelf MPSoCs. Decapping MPSoCs and observing them through an IR camera is possible [30], but it prevents connecting the chip to the desired heat dissipation solution.

For these reasons, the validation of the thermal model has been performed using a dedicated thermal test platform [11]. The chip is shown in Fig. 4. This platform is built around a custom silicon integrated circuit providing a 4x4 array of

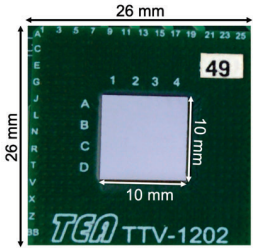

Fig. 4. The thermal test chip used for the validation.

heating elements and temperature sensors, as well as support electronics to apply arbitrary power profiles and measure temperatures with a 0.1° resolution.

B. Heat Sink and Fan

We use an HS483-ND copper heat sink and P14752-ND fan whose operating voltage ranges from 12 V to 27.6 V. We assume 12 V, 18 V, and 24 V as the three available fan voltages. Therefore, by using the Fan Laws, we relate the fan speed to its power consumption, as follows:

$$P_{fan,k} = P_{fan,l} \times (\frac{S_{fan,k}}{S_{fan,l}})^3, \tag{5}$$

where P_{fan} and S_{fan} are fan power consumption and fan speed, respectively.

Although P14752-ND is a low-power small fan suitable for the TTC described in Section V-A, our work is valid for any type of fan with any range of operating voltage/speed and chip. Therefore, we would release it as an open-source thermal simulator.

C. Power and Performance Model

In this work, we assume that the performance of an application is equivalent to the speedup obtained compared to a baseline performance, which we consider when the application is running on a single core with the minimum operating frequency. Therefore, we consider the modified Amdahl's Law [31], where the speedup also linearly scales with the ratio of current operating frequency over minimum frequency (i.e., in our case, F/F_{min}):

$$perf = \frac{1}{\frac{f}{n} + (1-f)} \times \frac{F}{F_{min}} \tag{6}$$

In this equation, n is the number of cores, $0 \leq f \leq 1$ represents the fraction of the application that can be run in parallel. In our model, we assume $f = 1$. Hence, for the experiments, we generate a baseline dynamic power trace with random fluctuation between 0.2 to 1.2 Watts (i.e., dynamic power of a single core operating under the minimum frequency). The execution time and power consumption are, thus, scaled according to the system configuration following (6). Moreover, we assume that all cells, representing a processing core, have a static power consumption of 0.3 Watts. Our assumptions in the power and performance model make static power, dynamic

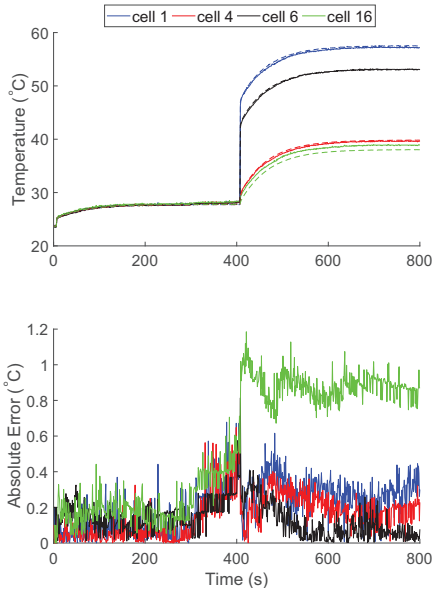

Fig. 5. Validation of transient fan model validation

TABLE II
STD AND MAE FOR DIFFERENT POWER MAPPINGS

	4 cores	8 cores	16 cores
MAE ($^\circ C$)	0.24	0.30	0.47
STD ($^\circ C$)	0.31	0.23	0.42
Max. Error ($^\circ C$)	1.2	1.1	1.9

power value and variation frequency within the range of those of commercial boards, such as, Xilinx Zynq boards[1].

VI. Experimental Results and Discussion

A. Validation: Transient Fan Model

Fig. 5 shows the temperature measured from the TTC per-cell sensors (solid line), simulated by our model (dashed), and the corresponding error. For these plots, all cells consume 0.2 W at the beginning. Then, at $t = 400sec$, four top-left cores are powered up to 3 W while the rest remain at 0.2 W. In addition, we validate our model considering two more power maps where I) all sixteen cores, and II) eight cores in a checkerboard form are powered up to 3 W at $t = 400sec$. Note that, in this figure, we show the temperature of four cells, but similar plots can be shown for other cells. Finally, the maximum absolute error, standard deviation (STD), and the mean absolute error (MAE) of the simulated temperature with respect to the measurements for different power mappings are tabulated in TABLE II. As shown in the table, the maximum error is below $2^\circ C$.

B. Thermal Management Evaluation

As explained in Section IV, we let the QL agent explore the design space and learn how to apply a sequence of actions to meet predefined objectives and constraints through a simulation framework. In our case of study, the agent learns to use all available frequency and fan speed values with a subset of available number of cores and the corresponding mapping. In particular, the QL agent learns to discard using 1_{corner}, 2_{corner}, 4_{row}, and 8_{row}, since these actions are either unable to maximize the performance, or may result in thermal violation if applied in specific states. Once exploration and exploration-exploitation phase finish on the simulation framework, the QL agent is ready to fully exploit its knowledge on a real hardware, in our case, the TTC.

[1]https://www.xilinx.com/products/technology/power/xpe.html

TABLE III
COMPARISON BETWEEN DIFFERENT POLICIES AT TWO THERMAL CONSTRAINTS

θ_{crit}		$\pi_{QL,adaptive}$	$\pi_{QL,max}$	π_{soa}
$98^\circ C$	#Violations/Duration (s)	0/0.0	0/0.0	2/10.8
	Normalized Fan Power	0.60	1.0	0.60
	Execution Time (s)	4790	4740	5309
	Average Hot Spot ($^\circ C$)	74.9	70.8	73.8
$80^\circ C$	#Violations/Duration (s)	0/0.0	0/0.0	7/39.2
	Normalized Fan Power	0.73	1.0	0.76
	Execution Time (s)	5073	4900	6228
	Average Hot Spot ($^\circ C$)	73.1	67.7	70.5

To show how the proposed QL-based DTM with proactive fan speed control can enhance performance with minimized fan power, we implement two more thermal management policies that maximize performance objective in addition to our proposed policy, $\pi_{QL,adaptive}$. In $\pi_{QL,max}$, we implement the same QL-based approach while limiting the available actions to only frequency and number of cores with the fan speed fixed at the maximum speed. In π_{soa}, we implement a reactive fan speed control policy similar to the default configuration explained in [32]. In this policy, the fan speed is initially set to the minimum value unless the two predefined thermal thresholds are violated. In such cases, the next two larger fan speeds are used. Finally, for performance maximization, we adapt the proposed solution by [21].

TABLE III compares the number and duration of thermal violations, normalized fan power, execution time representing the performance, and the average hot spot temperature for two different thermal constraints, θ_{crit}. As shown in the table, our approach achieves a DTM policy through which the fan power is reduced by 40% and 27%, respectively when $\theta_{crit} = 98^\circ C$ and $\theta_{crit} = 80^\circ C$, compared to the maximum fan policy, with only 1% performance degradation and no thermal violations. Note that with a lower θ_{crit}, $\pi_{Ql,adaptive}$ has to use higher fan speeds more frequently to avoid thermal violations while maximizing the performance. Compared to the state-of-the-art reactive DTM approach, our solution, as shown in the table, is able to improve the performance 11%, and 19% for the two thermal constraints without any further fan power and, yet, with no thermal violations. π_{soa} causes thermal constraint violations twice since, opposed to $\pi_{QL,adaptive}$ and $\pi_{QL,max}$, it cannot foresee the temperature increase resulting from changes in operating frequency, number of active cores, etc., regarding a particular initial temperature or thermal history. This information, on the contrary, can be learned by RL.

Fig. 6 shows the first 300 seconds of the hot spot and fan speed trace obtained from different implemented policies sampled at 100 Hz. Each value in these plots could belong to any of the 16 cells on the TTC. As shown in the figure,

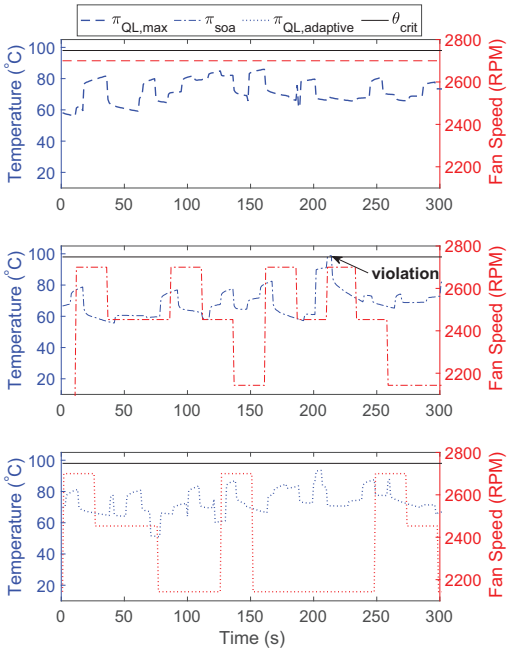

Fig. 6. Hot spot and fan speed comparison

while $\pi_{QL,adaptive}$ and $\pi_{QL,max}$ do not cause any thermal violations, π_{soa} is unable to avoid thermal violations. For a lower thermal constraints, more violations occur by π_{soa}, i.e., seven for a total duration of 39.2 seconds. In fact, the TTC experience delayed hot spot alleviation due to the reactive behavior of π_{soa}. In contrast, $\pi_{QL,adaptive}$ is able to proactively tune the fan speed for each thermal constraint.

VII. Conclusion

In this work, we first proposed and integrated a fan model for transient simulation into the open-source state-of-the-art thermal simulator, 3D-ICE. We validated our model on a thermal test chip with less than $2°C$ for the worst-case scenario. Since our model would be valid for any type of fan and chip, we will release it as an open-source thermal simulator. Second, we proposed a reinforcement learning based approach for dynamic thermal management of multiprocessor platforms with proactive fan speed control for performance maximization and fan power minimization under thermal constraints. Our results showed up to 40% decrease in fan power consumption when compared to state-of-the-art DTM policy with a fixed fan speed, and up to 19% improvement in performance with no further fan power compared to a state-of-the-art reactive DTM policy.

References

[1] A. K. Coskun *et al.*, "Temperature aware task scheduling in MPSoCs," in *DATE*. IEEE, 2007.

[2] A. Iranfar *et al.*, "Thespot: Thermal stress-aware power and temperature management for multiprocessor systems-on-chip," *IEEE TCAD*, 2017.

[3] A. Naveh *et al.*, "Power and thermal management in the intel core duo processor." *Intel Technology Journal*, vol. 10, no. 2, 2006.

[4] D. Brooks and M. Martonosi, "Dynamic thermal management for high-performance microprocessors," in *HPCA*. IEEE, 2001.

[5] F. Terraneo *et al.*, "A cycle accurate simulation framework for asynchronous NoC design," in *2013 International Symposium on System on Chip (SoC)*, 2013, pp. 1–8.

[6] Z. Jian-Hui and Y. Chun-Xin, "Design and simulation of the CPU fan and heat sinks," *IEEE Transactions on Components and Packaging Technologies*, 2008.

[7] D. Hanrahan, "Fan-speed control techniques in PCs," *Analog Dialogue*, 2000.

[8] Z. Wang *et al.*, "Optimal fan speed control for thermal management of servers," in *ASME 2009 InterPACK Conference*. American Society of Mechanical Engineers, 2009.

[9] A. Sridhar *et al.*, "3D-ICE: Fast compact transient thermal modeling for 3D ICs with inter-tier liquid cooling," in *ICCAD*. IEEE, 2010. [Online]. Available: https://esl.epfl.ch/research/page-26259-en-html/page-42393-en-html/3d-ice-html/3d-ice-download/

[10] M. Zapater *et al.*, "Leakage and temperature aware server control for improving energy efficiency in data centers," in *DATE*. EDA Consortium, 2013.

[11] F. Terraneo *et al.*, "An Open-Hardware Platform for MPSoC Thermal Modeling," in *International Conference on Embedded Computer Systems*. Springer, 2019.

[12] Q. Xie *et al.*, "Therminator: a thermal simulator for smartphones producing accurate chip and skin temperature maps," in *ISLPED*. ACM, 2014.

[13] S. Ladenheim *et al.*, "IC thermal analyzer for versatile 3-D structures using multigrid preconditioned Krylov methods," in *ICCAD*. IEEE, 2016.

[14] W. Huang *et al.*, "HotSpot: A compact thermal modeling methodology for early-stage VLSI design," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 2006.

[15] D. Shin *et al.*, "Energy-optimal dynamic thermal management: Computation and cooling power co-optimization," *IEEE Transactions on Industrial Informatics*, 2010.

[16] M. M. Sabry and D. Atienza, *Temperature-Aware Design and Management for 3D Multi-Core Architectures*, 2014.

[17] T. Chantem *et al.*, "Temperature-aware scheduling and assignment for hard real-time applications on MPSoCs," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 2010.

[18] J. C. Salinas-Hilburg *et al.*, "Unsupervised power modeling of co-allocated workloads for energy efficiency in data centers," in *DATE*. IEEE, 2016.

[19] M. J. Dousti and M. Pedram, "Power-aware deployment and control of forced-convection and thermoelectric coolers," in *DAC*. IEEE, 2014.

[20] W. Zheng *et al.*, "TECfan: Coordinating Thermoelectric Cooler, Fan, and DVFS for CMP Energy Optimization," in *IPDPS*. IEEE, 2016.

[21] V. Hanumaiah and S. Vrudhula, "Energy-efficient operation of multicore processors by dvfs, task migration, and active cooling," *IEEE TC*, 2012.

[22] B. Acun *et al.*, "Neural network-based task scheduling with preemptive fan control," in *Proceedings of the 4th International Workshop on Energy Efficient Supercomputing*. IEEE Press, 2016.

[23] F. Terraneo *et al.*, "Event-based thermal control for high power density microprocessors," in *Harnessing Performance Variability in Embedded and High-performance Many/Multi-core Platforms*. Springer, 2019.

[24] C. S. Chan *et al.*, "Optimal performance-aware cooling on enterprise servers," *IEEE TCAD*, 2018.

[25] J. Kim *et al.*, "Global fan speed control considering non-ideal temperature measurements in enterprise servers," in *DATE*. European Design and Automation Association, 2014.

[26] P. Fritzson, *Principles of object-oriented modeling and simulation with Modelica 2.1*. John Wiley & Sons, 2010.

[27] P. Fritzson *et al.*, "Openmodelica - a free open-source environment for system modeling, simulation, and teaching," in *2006 IEEE Conference on Computer Aided Control System Design*, 2006, pp. 1588–1595.

[28] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press Cambridge, 1998, vol. 1, no. 1.

[29] A. Iranfar *et al.*, "A heuristic machine learning-based algorithm for power and thermal management of heterogeneous mpsocs," in *ISLPED*. IEEE, 2015, pp. 291–296.

[30] H. Amrouch and J. Henkel, "Lucid infrared thermography of thermally-constrained processors," in *ISLPED*, July 2015.

[31] D. H. Woo and H.-H. S. Lee, "Extending amdahl's law for energy-efficient computing in the many-core era," *Computer*, 2008.

[32] G. Singla *et al.*, "Predictive dynamic thermal and power management for heterogeneous mobile platforms," in *DATE*. EDA Consortium, 2015.