Enhancing Manycore Lifetime Through Reinforcement Learning Task Mapping and Migration

Iaçanã Ianiski Weber, Vitor Balbinot Zanini, Fernando Gehm Moraes
School of Technology, Pontifical Catholic University of Rio Grande do Sul – PUCRS – Porto Alegre, Brazil
iacana.weber@pucrs.br, vitor.balbinot@edu.pucrs.br, fernando.moraes@pucrs.br

***Abstract*—Manycore systems emerged as a solution to the limitations of single-core processors in meeting modern computational demands. Effective task mapping and migration are essential in these systems to optimize computational performance without exceeding the Thermal Design Power (TDP) constraints. Additionally, temperature management is crucial for ensuring the system's long-term reliability. This research proposes a lightweight and scalable heuristic for reliability-aware task mapping and migration. Our approach employs machine learning techniques, specifically Reinforcement Learning (RL), to optimize system mapping and migration. The proposed method utilizes a lookup table, which is pre-trained using Q-learning. The pre-training enables dynamic task distribution adjustments in response to the task mapping and their power consumption. Experimental results demonstrate that it outperforms other strategies. Our proposed method effectively manages peak temperatures and improves the system's Mean Time To Failure (MTTF). This study provides a robust framework for task management in manycore systems and sets the groundwork for future explorations into autonomous system optimization.**

***Index Terms*—Manycore, Reliability, Reinforcement Learning, Task Mapping, Task Migration.**

I. Introduction

An efficient way of managing manycore systems involves task mapping and migration, essential for optimizing resource use by enhancing system performance and thermal behavior [1]. Such management aims to maximize workload distribution into the system Processing Elements (PEs) while managing the inactive part of the silicon, so-called *dark* regions [2].

Managing the manycore temperature is crucial for enhancing system lifetime reliability [3]. Several proposals emerged in the literature utilizing machine learning techniques, particularly Reinforcement Learning (RL), for Dynamic Thermal Management (DTM). Das et al. [4] employ Q-learning to build a Q-table that indexes system states representing all possible task arrangements within the system. However, a significant challenge arises due to the exponential growth of the action space as the system size increases, making their proposal unscalable.

In contrast, Rathore et al. [5] present an alternative approach by creating a Q-table with a fixed action space. They classify applications based on power, temperature, and computation requirements. RL is used to train the Q-table for optimal decision-making regarding whether an application should run on a high or low-frequency core to maximize system health. Nonetheless, Rathore's method relies on physical embedded sensors in each core to provide real-time data to the learning algorithm, potentially limiting scalability by requiring these hardware components.

Zhang et al. [6] propose a multi-layer perceptron (MLP) neural network that can predict PE temperature. The MLP is trained using Intel's Performance Counter Monitor and thermography data. Using the estimations, the authors can distribute tasks to preserve core reliability and manage temperature. However, the MLP requires several milliseconds to estimate temperature, creating a performance drawback for each task allocation.

Limitations of previous work motivate the *goal* of our work: *propose a lightweight and scalable reliability-aware mapping algorithm that operates independently of physical sensing data*. To achieve this goal, our proposal is based on three key considerations, detailed in the text:

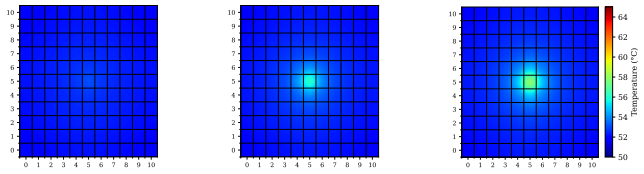
I. The heating behavior within a specific PE results mainly from the power it dissipates and the thermal heat from neighboring PEs.
II. The numerous wear-out effects are intricately linked to the system's temperature.
III. The use of RL as a scalable approach to establish a connection between power consumption and task mapping, focusing on enhancing system reliability.

This paper is structured as follows. Section II discusses the heating behavior of a PE and its impact on reliability. Section III introduces our original contribution, detailing the training process. Section IV describes the experimental setup, including simulated scenarios and data to ensure reproducibility. Section V presents results in terms of temperature and reliability. Section VI concludes the paper and suggests future research directions.

II. Heat and Reliability

The heat generated by a specific PE during the execution of a particular task is a function of its power dissipation. We periodically estimate the energy consumption of each PE, which results in its power dissipation. The energy is determined by three components: (i) the energy associated with memory accesses [7]; (ii) the energy related to communication with the NoC router and the transmission of packets through the PE [8]; (iii) the energy consumed by the RISC-V core while executing instructions [9]. Figure 1 illustrates the temperature after 20 seconds of executing three tasks on our manycore platform [10], (τa, τb, and τc) mapped to the central PE ($x = 5$, $y = 5$) within an 11x11 manycore. It is noteworthy

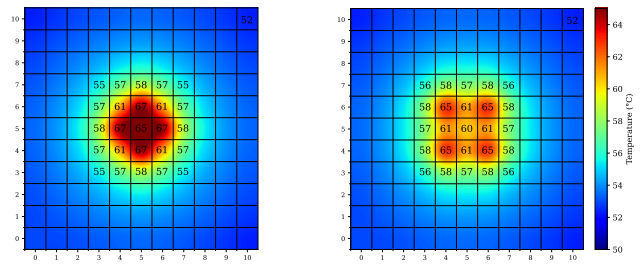
that the average power consumption of each task increases in the following order: $P_{\tau(a)} < P_{\tau(b)} < P_{\tau(c)}$. We observe progressively higher temperatures relative to surrounding PEs due to varying task power consumption.



(a) (b) (c)

Fig. 1: Thermal shots of an 11x11 manycore system. Each thermal map corresponds to the execution of τa (a), τb (b) and τc (c). The colors indicate the temperature of the PEs, with warmer colors corresponding to higher temperatures.

The heat generated by PEs is transferred through thermal conduction to adjacent PEs. This phenomenon is illustrated in Figure 2, which depicts four τc tasks allocated in an 11x11 manycore. These tasks are positioned around the central PE in two distinct configurations: (a) $\{[6,5],[5,4],[5,6],[4,5]\}$ with tasks mapped orthogonally to the central PE, and (b) $\{[4,4],[4,6],[6,6],[6,4]\}$ with tasks mapped diagonally to the central PE. A temperature difference of 5.25°C in the central PE is observed between these two scenarios. This difference is attributable to the heat conducted from the four tasks surrounding the central PE. In scenario (a), these tasks create a contiguous area in contact with the central PE, facilitating significant heat transfer. Conversely, in scenario (b), where the tasks are diagonally positioned relative to the central PE, the area of contact is reduced, leading to a comparatively lower temperature increase. *This observation motivates the exploration of techniques to optimize the topological distribution of tasks within the manycore architecture to enhance its thermal performance.*



(a) (b)

Fig. 2: Thermal shot of an 11x11 manycore system executing four tasks. In (a), the four tasks are orthogonal to central PE. In (b), the four tasks are diagonal to the central PE. The colors indicate the temperature of the PEs, with warmer colors meaning higher temperatures.

We consider five wear-out effects related to the Arrhenius relationship [11], with models embedded in our framework. The considered effects are: (*i*) Electromigration (EM), (*ii*) Stress Migration (SM), (*iii*) Time-Dependent Dielectric Breakdown (TDDB), (*iv*) Thermal Cycling (TC), and (*v*) Negative Bias Temperature Instability (NBTI). Each effect is characterized by a Failure-in-Time (FIT) equation in which temperature directly influences the wear-out rate [12].

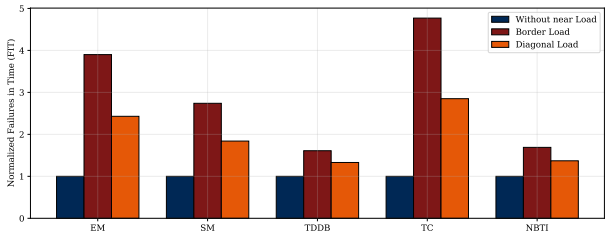


Fig. 3: FIT estimation for the central PE ($x = 5$, $y = 5$), in three different conditions: (*i*) without any task being executed near to it; (*ii*) mapping of Figure 2(a); (*iii*) mapping of Figure 2(b). Simulated on [10].

Figure 3 shows the normalized FIT considering all wear-out effects over the central PE in three different scenarios. It is important to highlight that the probed PE remains idle in the three simulations. The Figure shows that the increase in temperature directly impacts the number of failures experienced by the device. The effect of the temperature increase varies, with a 28% increment for TDDB and a 192% increase for TC when comparing scenarios (*ii*) and (*iii*). The increase in reliability degradation is due to the heat generated by the temperature of nearby PEs. However, estimating the FIT at runtime may represent a significant cost to the manycore manager as every effect must be computed to each PE at every monitoring window. *Even with FIT information available, to our knowledge, there is no management technique available that considers the FIT value in its cost function.*

III. Reinforcement Learning-based Task-to-Core Heuristics

This section introduces the original contribution of this paper. It employs the RL method, called Q-learning, which was selected for its simplicity and model-free characteristic, meaning it operates without requiring a model of the environment. Comparative analysis reveals that Q-learning outperforms other model-free RL algorithms in key metrics: it has the lowest Root Mean Square (RMS) error, highest speed of convergence, and largest average total reward return [13].

In our approach, we added a lookup table in the system manager PE named Q-table. The Q-learning method populates the Q-table at pre-runtime. The mapping and migration heuristics use the Q-table values (named Q-values) at runtime.

The Q-table is a $r \times c$ matrix used by the task mapping and migration heuristics. Each row is addressed by the task power dissipation, or "task power category" (**TPC**). A k-means clustering algorithm discretizes every task into three groups at pre-runtime according to their power dissipation. Each column is indexed by a value defined by the thermal influence of the neighbors of a given PE, named "PE bin-state" (**PBS**). The total number of PBS is equal to 140. The PBS is an index in a table that differentiates the distribution of tasks around a given PE. Exemplifying, we say that one PE neighbored only by idle PEs is said to be in the PBS = 0 (Figure 1). While the same PE, when surrounded by 4 PEs executing high computational demanding tasks (TPC = 2), like in the scenario in Figure 2(a), we say that the PE is in the PBS = 4. However, when considering the same tasks disposed diagonally to the probed PE, like Figure 2(b), the PBS = 39. We have 140 different possible TPC arrangements around a given PE. The Q-learning training phase determines the $r \times c$ Q-values.

During pre-runtime, we perform the training phase, where the Q-table is initialized with high values to enable the state-space exploration. The training process is depicted in Figure 4. Three processes are parallel (*blue*, *orange*, and *red* paths, from the Figure 4). One of them is triggered when an application arrives randomly in the system. Every application task is mapped using the epsilon-greedy allocation (to increase the space exploration during the training) [14]. Two parallel processes are triggered every $1ms$ (monitoring window). The temperature and FIT are estimated by a hardware accelerator [15] and sent to the system manager (*red* path) . The other process is the Q-learning loop executed for each occupied PE that is stable, i.e. no changes in the PBS for at least $50ms$ (empirically obtained). If the PE is stable, one reward is given to the PE based on the FIT variation during the past $50ms$. The Q-learning equation [16] is used to update a Q-value in the Q-table based on the PE PBS/TPC and reward. When the learning rate reaches zero ($\alpha = 0$), the simulation stops as the pre-runtime step is concluded, and the Q-table is ready to be used at runtime to execute workloads using the learned Q-table in the mapping and migration heuristics.

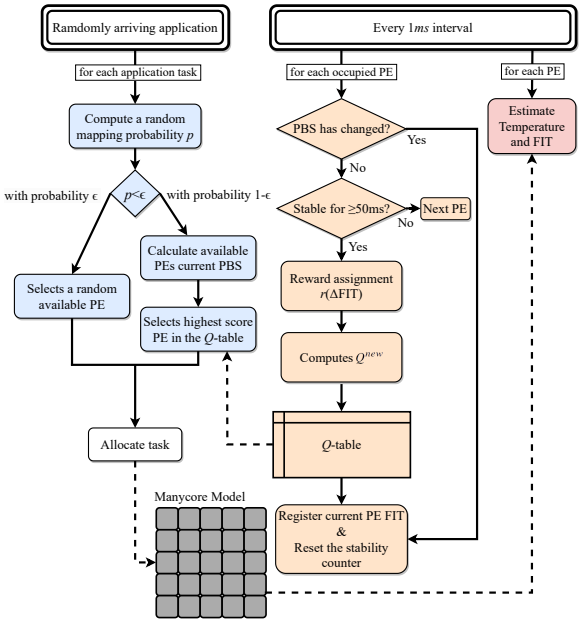


Fig. 4: Pre-runtime training loop [17].

A. Application Admission

Application admission is a process executed by the manager PE triggered by a request to execute a new application into the system. This process searches for a cluster that contains resources for the incoming application. The application is admitted if a cluster has enough resources, potentially spanning the entire system when resources are scarce. The cluster selection aims to optimize the task placement, minimizing communication delays and congestion by reducing the hop count between communicating tasks. The algorithm uses a sliding search window method using the temperature of the PEs inside the cluster and the number of available resources to execute the tasks, selecting the cluster with the lowest temperature.

B. Mapping Heuristic

The task mapping starts after the application admission step and is repeated for each application task. The manager PE executes the mapping heuristic using the Q-table. One loop will iterate over every PE within the selected cluster. When a given PE is available, its score is obtained by consulting the Q-table, indexed by the PBS and the TPC. If the PE score exceeds the prior highest score, then the highest score is the target PE. If more than one PE presents the same highest Q-value, the heuristics selects the PE that minimizes the hop count between tasks. After repeating this verification for each PE in the cluster, the algorithm returns the PE with the highest Q-table score. Therefore, our mapping algorithm is $O(len(n))$ complex, with $len(n)$ representing the size of the cluster.

C. Migration Heuristic

Task migration serves as an actuation mechanism to lower temperatures in specific regions of a manycore system. This process is initiated at each monitoring interval. The migration heuristic continuously scans for thermal violations by comparing temperatures against a pre-defined thermal threshold, T_{th}. Upon identifying a thermal violation, the managing PE determines which PE has the highest temperature exceeding T_{th} and selects the task running on this PE for migration.

After selecting the task to migrate, the heuristic creates a list of all the PEs with a task slot available within an expanded application cluster. The PE list is sorted by temperature. Similarly to the task mapping process, the score is obtained by consulting the Q-table, indexed by the PBS and the TPC. The algorithm evaluates only the cooler half of the available PEs, looking for the one with the highest score. This method makes a trade-off between current temperature and Q-table score. If more than one PE presents the same highest score, the PE with the lowest temperature is selected.

Note that each migration implies an execution overhead for the application because the migration protocol requires multiple synchronization packets to transfer the task memory content to its new PE [18]. Thus, a mapping heuristic that minimizes the number of migrations is essential to optimizing application performance.

IV. Experimental Setup

We used a set of applications for the experiments: Dijkstra (DIJ), 7 tasks; Dynamic Time Warping (DTW), 6 tasks; Synthetic producer/consumer (PRCO), 2 tasks; Synthetic application α (ALPHA), 4 tasks; Synthetic application β (BETA), 4 tasks; Synthetic application γ (GAMMA), 8 tasks; Synthetic application double pipeline (DPIPE), 7 tasks.

Sixteen scenarios were created to evaluate the thermal and reliability behavior of different management techniques, presented in Table I. They were simulated in 8x8 and 14x14 manycores. Each PE contains a scratchpad memory, one router, a network interface with DMA capability [19], and a RISC-V 32IM core. Chronos-V is the reference manycore platform [10], modeled with OVP [20], based on the Memphis core [21]. The workload varies from computation-intensive (CI) to a mix of computation- and communication-intensive tasks (M1, M2, M) and varying occupancy (R). The scenarios also vary the system occupancy, i.e., the percentage of PEs in use at any

TABLE I: Evaluated scenarios.

Scenario				Applications							
Sys.	Workload	Occ. (%)	Tag	DIJ	DTW	PRCO	ALPHA	BETA	GAMMA	DPIPE	Total Tasks
8x8	M1	50	A5			4	3	3			32
		70	A7			6	5	4			48
		90	A9			7	7	4			58
	M2	50	B5	2		5	2				32
		70	B7	2	1	7	2	1			46
		90	B9	3	1	9	3				57
	CI	50	C5				8				32
		70	C7				9		1		44
		90	C9				10		2		56
14x14	M	50	D5	5		5		1		7	98
		70	D7	6		10		3		9	137
		90	D9	6		13		8		11	177
	CI	50	E5				24				96
		70	E7				34				136
		90	E9				44				176
	R	~	F	3	3	3	3	3	3	3	~

time during the execution, with levels ranging from 50% to 90%.

We evaluated each scenario running with four different management heuristics. Two are static mapping strategies, **grouped mapping**, and **pattern mapping** [22, 23]. The grouped mapping groups tasks to minimize the hop count. Meanwhile, the patterning mapping allocates tasks following a chessboard pattern, interleaving idle and active PEs. Following, we have the dynamic-thermal management based on a proportional integrative and derivative control loop (**PID**) [24]. The PID strategy periodically calculates a PID score for each PE. The heuristic selects the PE with the best PID score to map/migrate a task. Lastly, our management uses the ***Q*-table** to perform the mapping and migration heuristics.

V. Results

Table II compares our management technique with the Grouped, Pattern, and PID approaches. Table II(a) compares the **Average Peak Temperature** in °C. We observe that, on average, our heuristic reduced the average peak temperature in 6.96°C, 0,25°C, and 1,61°C when compared to Grouped, Pattern, and PID, respectively. The **Grouped** significantly underperforms against every other heuristic across all scenarios, with the greatest thermal disadvantage reaching 18.51°C over our proposal. The **Pattern** demonstrates mixed efficacy. Its main advantage arrives in scenarios with 50% occupation (A5, B5, C5, D5, E5), where the Pattern heuristic can interleave active and inactive PEs. However, this advantage diminishes as the occupancy increases. Besides our proposal, the **PID** is the only management that provides migration capabilities. However, the migration capabilities could not provide better results than Pattern, with scenarios reaching peak temperatures that surpassed 8ºC compared to our proposal.

TABLE II: Average Peak Temperature and MTTF comparison against our Proposal.

Average Peak Temperature Comparison (°C)

Heuristic	Scenarios																Average
	A5	A7	A9	B5	B7	B9	C5	C7	C9	D5	D7	D9	E5	E7	E9	F	
Grouped	-10,73	-4,72	-3,38	-6,22	-4,77	-1,95	-8,67	-7,37	-1,81	-9,77	-6,71	-5,45	-18,51	-9,62	-4,77	-7,04	**-6,96**
Pattern	1,96	0,56	-1,81	1,13	0,31	-0,35	2,95	-0,12	-1,08	1,59	-1,54	-1,22	-0,21	-3,48	-2,49	0,73	**-0,25**
PID	-3,10	-0,35	-0,41	-1,21	-0,65	-0,64	0,23	-1,61	-0,85	-2,75	-1,09	-0,75	-8,21	-2,57	-0,22	-1,46	**-1,61**

(a)

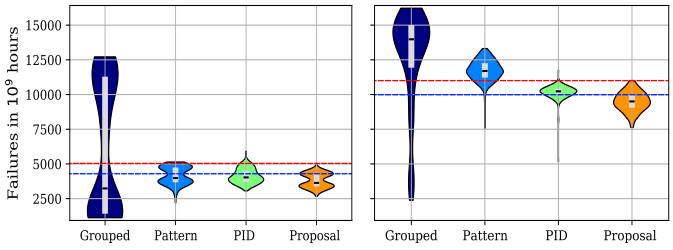
MTTF Comparison (%)

Heuristic	Scenarios																Average
	A5	A7	A9	B5	B7	B9	C5	C7	C9	D5	D7	D9	E5	E7	E9	F	
Grouped	26,39	12,33	14,25	14,25	11,16	5,87	31,18	28,44	14,12	19,95	17,61	16,96	48,64	42,94	28,04	17,52	**22,14**
Pattern	1,01	2,12	10,69	-0,47	-2,33	1,57	-0,38	7,57	13,82	1,81	0,31	1,06	8,49	17,06	19,63	2,96	**5,46**
PID	5,68	2,12	6,11	3,42	0,93	3,52	2,28	5,96	10,00	1,13	0,00	0,71	6,55	10,00	6,54	1,08	**4,33**

(b)

Table II(b) shows that our proposal improves the MTTF by an average of 22.14% over the **Grouped** heuristic. Compared to the **Pattern**, the improvement averages 5.46%, demonstrating that while our proposal is beneficial, the improvement varies by scenario. Specifically, the improvement ranges from a decrease of 2.33% in scenario B7 to an increase of 19.63% in E9. Against the **PID** heuristic, our proposal averages a 4.33% improvement in MTTF. This data suggests that temperature reductions are not the sole determinant of manycore lifetime, as evidenced by the PID's superior lifetime performance despite achieving a lesser temperature reduction than the Pattern.

Figure 5 provides a comprehensive view of the FIT distribution. This graphical representation depicts FIT values across violin-boxes that span from the minimum to the maximum recorded. The width of each box at different FIT levels indicates the density of PEs. The white bar within each box represents the interquartile range, extending from the first to the third quartile. A black dot in this bar marks the median FIT value, showing that half of the PEs have lower FIT values and half have higher. The Grouped mapping, aiming to minimize hop count, exhibits the poorest FIT distribution.



(a) (b)

Fig. 5: Violin-box plot showing FIT distribution for 14x14 computation intensive (a) 50% and (b) 90% occupancy scenarios, respectively.

In Figure 5(a), the top 25% of PEs, defined by the area between the blue and red lines, correspond to the highest FIT values in our proposal. We observe that nearly 50% of the PEs are located in this region for the Pattern and PID mappings, which explains the lower MTTF for these heuristics. In Figure 5(b), we observe a more pronounced difference in the FIT distribution. Over 75% of PEs under the Pattern mapping exhibited worse results compared to the maximum FIT in our proposal. Similarly, compared to the PID approach, over 75% of PEs demonstrated poorer FIT than the top 25% of our proposal. Such graphs demonstrate the effectiveness of the proposed RL in inproving the system lifetime.

VI. Conclusion

This work points in the direction of developing a lightweight and scalable reliability-aware mapping algorithm that operates independently of physical sensing data. The proposed heuristic uses a lookup table trained using the *Q*-learning algorithm to decide where in the manycore is the best place to allocate a task to maximize the system reliability. The results demonstrate our proposal's efficiency in maximizing MTTF and temperature control in scenarios with higher workloads and larger system dimensions. Future works encompass simulations in RTL-level platforms and the definition and validation of an online learning heuristic without requiring a pre-runtime phase.

ACKNOWLEDGMENTS

This work was financed in part by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), Finance Code 001; Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), 309605/2020-2 and 407829/2022-9; Fundação de Amparo à Pesquisa do Estado do Rio Grande do Sul (FAPERGS), 21/2551-0002047-4 and 23/2551-0002200-1.

REFERENCES

[1] B. Li, X. Wang, A. K. Singh, and T. Mak, "On Runtime Communication and Thermal-Aware Application Mapping and Defragmentation in 3D NoC Systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 12, pp. 2775–2789, 2019, https://doi.org/10.1109/TPDS.2019.2921542.

[2] A. Karkar, N. Dahir, T. Mak, and K.-F. Tong, "Thermal and performance efficient on-chip surface-wave communication for many-core systems in dark silicon era," *ACM Journal on Emerging Technologies in Computing Systems*, vol. 18, no. 3, pp. 49:1–49:18, 2022, https://doi.org/10.1145/3501771.

[3] V. Rathore, V. Chaturvedi, A. K. Singh, T. Srikanthan, and M. Shafique, "Longevity Framework: Leveraging Online Integrated Aging-Aware Hierarchical Mapping and VF-Selection for Lifetime Reliability Optimization in Manycore Processors," *IEEE Transactions on Computers*, vol. 70, no. 7, pp. 1106–1119, 2021, https://doi.org/10.1109/TC.2020.3006571.

[4] A. Das, R. A. Shafik, G. V. Merrett, B. M. Al-Hashimi, A. Kumar, and B. Veeravalli, "Reinforcement learning-based inter-and intra-application thermal optimization for lifetime improvement of multicore systems," in *ACM/IEEE Design Automation Conference (DAC)*, 2014, pp. 1–6, https://doi.org/10.1145/2593069.2593199.

[5] V. Rathore, V. Chaturvedi, A. K. Singh, T. Srikanthan, and M. Shafique, "Towards Scalable Lifetime Reliability Management for Dark Silicon Manycore Systems," in *IEEE International Symposium on On-Line Testing and Robust System Design (IOLTS)*, 2019, pp. 204–207, https://doi.org/10.1109/IOLTS.2019.8854454.

[6] J. Zhang, S. Sadiqbatcha, and S. X. Tan, "Hot-Trim: Thermal and Reliability Management for Commercial Multicore Processors Considering Workload Dependent Hot Spots," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 42, no. 7, pp. 2290–2302, 2023, https://doi.org/10.1109/TCAD.2022.3216552.

[7] S. Li, K. Chen, J. H. Ahn, J. B. Brockman, and N. P. Jouppi, "CACTI-P: Architecture-level modeling for SRAM-based structures with advanced leakage reduction techniques," in *IEEE International Conference on Computer-Aided Design (ICCAD)*, 2011, pp. 694–701, https://doi.org/10.1109/ICCAD.2011.6105405.

[8] A. L. Martins, A. H. L. da Silva, A. M. Rahmani, N. Dutt, and F. G. Moraes, "Hierarchical adaptive Multi-objective resource management for many-core systems," *Journal of Systems Architecture*, vol. 97, pp. 416–427, 2019, https://doi.org/10.1016/j.sysarc.2019.01.006.

[9] G. Y. L. Fang, "Instruction-Level Power Consumption Simulator for Modeling Simple Timing and Power Side Channels in a 32-bit RISC-V Micro-Processor," Master's thesis, Massachusetts Institute of Technology (MIT), 2021, 140p.

[10] I. I. Weber, A. Dal Zoto, and F. G. Moraes, "Chronos-V: a Many-core High-level Model with Support for Management Techniques." *Analog Integrated Circuits and Signal Processing*, vol. 117, p. 57–71, 2023, https://doi.org/10.1007/s10470-023-02190-8.

[11] S. Arrhenius, "Uber die Reaktionsgeschwindigkeit bei der Inversion von Rohrzucker durch Säuren," *Zeitschrift für physikalische Chemie*, vol. 4, no. 1, pp. 226–248, 1889, https://doi.org/10.1515%2Fzpch-1889-0416.

[12] J. Srinivasan, S. V. Adve, P. Bose, J. Rivers, and C.-K. Hu, "RAMP: A model for reliability aware microprocessor design," IBM, Tech. Rep., 2003, 26p.

[13] S. E. Li, *Reinforcement learning for sequential decision and optimal control*, 1st ed. Springer, 2023, 492p.

[14] W. B. Powell, *Reinforcement Learning and Stochastic Optimization: A Unified Framework for Sequential Decisions*, 1st ed. John Wiley & Sons, 2022, 1136p.

[15] A. L. d. Silva, A. L. d. M. Martins, and F. G. Moraes, "Fine-grain Temperature Monitoring for many-core Systems," in *IEEE Symposium on Integrated Circuits and Systems Design (SBCCI)*, 2019, pp. 1–6, https://doi.org/10.1145/3338852.3339841.

[16] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, pp. 279–292, 1992, https://doi.org/10.1007/BF00992698.

[17] I. I. Weber, V. B. Zanini, and F. G. Moraes, "FLEA - FIT-Aware Heuristic for Application Allocation in Many-Cores based on Q-Learning,," in *Brazilian Symposium on Computing Systems Engineering (SBESC)*, 2023, pp. 1–6, https://doi.org/10.1109/SBESC60926.2023.10324296.

[18] S. Bertozzi, A. Acquaviva, D. Bertozzi, and A. Poggiali, "Supporting task migration in multi-processor systems-on-chip: a feasibility study," in *IEEE Design, Automation Test in Europe Conference (DATE)*, vol. 1, 2006, pp. 1–6, https://doi.org/10.1109/DATE.2006.243952.

[19] M. Ruaro, F. Lazzarotto, C. Marcon, and F. Moraes, "DMNI: A specialized network interface for NoC-based MPSoCs," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2016, pp. 1202–1205, https://doi.org/10.1109/ISCAS.2016.7527462.

[20] Imperas, "Open Virtual Platforms - the source of Fast Processor Models & Platforms," 2021, http://www.ovpworld.org/, March 2024.

[21] M. Ruaro, L. L. Caimi, V. Fochi, and F. G. Moraes, "Memphis: a framework for heterogeneous many-core SoCs generation and validation," *Design Automation for Embedded Systems*, vol. 23, no. 3-4, pp. 103–122, 2019, https://doi.org/10.1007/s10617-019-09223-4.

[22] L. Yang, W. Liu, W. Jiang, M. Li, P. Chen, and E. H.-M. Sha, "Fotonoc: A folded torus-like network-on-chip based many-core systems-on-chip in the dark silicon era," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 7, pp. 1905–1918, 2017, https://doi.org/10.1109/TPDS.2016.2643669.

[23] W. Liu, L. Yang, W. Jiang, L. Feng, N. Guan, W. Zhang, and N. D. Dutt, "Thermal-aware Task Mapping on Dynamically Reconfigurable Network-on-Chip based Multiprocessor System-on-Chip," *IEEE Transactions on Computers*, vol. 67, no. 12, pp. 1818 – 1834, 2018, https://doi.org/10.1109/TC.2018.2844365.

[24] A. L. d. Silva, A. L. del Mestre Martins, and F. G. Moraes, "Mapping and Migration Strategies for Thermal Management in Many-Core Systems," in *IEEE Symposium on Integrated Circuits and Systems Design (SBCCI)*, 2020, pp. 1–6, https://doi.org/10.1109/SBCCI50935.2020.9189933.