CAnDy-TM: Comparative Analysis of Dynamic Thermal Management in Many-Cores using Model Checking

Syed Ali Asadullah Bukhari* , Faiq Khalid Lodhi*, Osman Hasan*, Muhammad Shafique† and Jörg Henkel‡
*School of Electrical and Computer Science, National University of Sciences and Technology, Islamabad, Pakistan
Email: {ali.asadullah, faiq.khalid, osman.hasan}@seecs.edu.pk
† Institute of Computer Engineering, Vienna University of Technology, Vienna, Austria
Email: muhammad.shafique@tuwien.ac.at
‡Chair for Embedded Systems, Karlsruhe Institute of Technology, Karlsruhe, Germany
Email: henkel@kit.edu

Abstract—**Dynamic thermal management (DTM) techniques based on task migration provide a promising solution to mitigate thermal emergencies and thereby ensuring safe operation and reliability of Many-Core systems. These techniques can be classified as central or distributed on the basis of a central DTM controller for the whole system or individual DTM controllers for each core or set of cores in the system, respectively. However, having a trustworthy comparison between central (c-) and distributed (d-) DTM techniques to find out the most suitable one for a given system is quite challenging. This is primarily due to the systemic difference between cDTM and dDTM controllers, and the inherent non-exhaustiveness of simulation and emulation methods conventionally used for DTM analysis. In this paper, we present a novel methodology called CAnDy-TM (stands for Comparative Analysis of Dynamic Thermal Management) that employs Model Checking to perform formal comparative analysis for cDTM and dDTM techniques. We identify a set of generic functional and performance properties to provide a common ground for their comparison. We demonstrate the usability and benefits of our methodology by comparing state-of-the-art cDTM and dDTM techniques, and illustrate which technique is good w.r.t. thermal stability and other task migration parameters. Such an analysis helps in selecting the most appropriate DTM for a given chip.**

I. Introduction

Coping with the high chip temperatures in Many-Core architectures is one of the growing concerns in the nano-scale era [1]. This issue has a direct impact on the reliability of Many-Core systems as excessive heat generation may lead to malfunctioning and physical damage of the chip. Various DTM techniques [2] have been proposed to mitigate this problem. The low-level DTM methods, like Dynamic Frequency and Voltage Scaling (DVFS), overcome the heating problem by slowing down the processor speed and thus compromising on performance [3]. However, managing the thermal issues by dynamic reallocation of the cores' utilization [4], [5], [6] has been found to be more performance efficient.

DTM techniques can be further categorized as *central* or *distributed* depending on the nature of the DTM controller [7]. In central DTM (cDTM), the centralized controller [8], [6] is responsible for managing the power hungry tasks evenly across the chip, and therefore, it needs to be aware of the state of all the cores. On the other hand, in distributed DTM (dDTM) [4], [5], each core or a set of cores is controlled by a dedicated controller (or agent) and the complete system comprises of several agents communicating only with the neighboring agents. The DTM agents regulate the chip temperature by invoking a task migration algorithm based on certain temperature threshold for a system. This task migration decision can either be *proactive* [9], [6], or *reactive* [4], [8].

Conventionally, simulation and emulation techniques are employed to analyze these techniques individually [10], [11]. However, these methods can analyze only a limited number of all the possible scenarios owing to a large number of DTM design parameters and the limited computational resources and thus result in a compromised analysis in terms of accuracy and completeness. Such limitations in the analysis may in turn lead to delays in deployment of DTM schemes as happened in the case of the Foxton DTM that was designed for the Montecito chip [12]. *Moreover, the variety of DTM design parameters and the limitations of conventional analysis methods complicate the task for comparing different DTM techniques and identifying the most effective one in terms of performance for a given scenario becomes very challenging.*

In the past couple of decades, the use of formal methods has been proposed to overcome and complement the limitations of simulation and emulation methods for DTM techniques [13], [14], [8]. More specifically, model checking, which is a state-space -based analysis technique, has been employed to analyse some of the state-of-the-art DTM techniques, like Thermal-aware Agent-based Power Economy (TAPE) [8], [15], [16]. In [8], the analysis of TAPE has been done for both functional and timing properties for a 3×3 core grid. However, during the analysis, the continuous parameters, like temperature, were abstracted using a small set of integer values and the number of cores could not be increased because of the infamous state-space explosion problem [17] of model checking. A solution to this problem has been proposed in [18] by using continuous-valued parametric modeling support in the nuXmv model [19] checker for analyzing a DTM-based task migration algorithm [4] for a 9×9 core grid. Probabilistic model checking has also been used for formal analysis of TAPE in [15] and [16], for 3×3 and 9×9 grid sizes, respectively. The results showed that the TAPE achieved thermal stability but did not perform the temperature balancing across the chip. In a recent work [20], a Formal Analysis Methodology for Task Migrations, i.e., FAMe-TM, has been proposed for dDTMs using nuXmv. FAMe-TM identifies a set of properties based on task migration principles and provides a generic methodology for formal modeling and analysis of task migration based dDTMs. In the above-mentioned works, individual analyses for given cDTM or dDTM techniques are provided, *but a common analysis framework applicable to both is missing.* Moreover, a comparison between cDTM and dDTM, based on communication overhead, which can be a key factor in performance evaluations of DTM techniques, is also missing. In the current paper, we aim at overcoming these deficiencies and bridging these gaps to provide a robust comparative framework. **Our Novel Contributions:** The novel features of CAnDy-TM include:

1) *Identification of a set of generic functional and performance properties* that can be formally verified for comparing cDTM and dDTM techniques.

2) *A step-wise procedure for developing formal models of both cDTM and dDTM techniques* and the ability to conduct an exhaustive comparative analysis, due to the completeness of model checking [17].

3) *Open-source code for CAnDy-TM, and the nuXmv models for state-of-the-art DTM schemes are available at [21].* This will help the DTM community to reproduce comparative results and design verification-friendly DTM techniques.

II. Proposed Methodology - CAnDy-TM

The first key step in the comparative analysis of DTM schemes using the proposed methodology, i.e., CAnDy-TM, as depicted in Fig. 1, is the modeling of a Many-Core grid system. Each core in the system is modeled as an individual module including the performance parameters that each core updates using counters for these parameters. These individual modules are then instantiated to compose an overall grid. The behavior of a cDTM or dDTM scheme is modeled by creating modules for the central and the distributed agents, respectively. In case of a cDTM scheme, one of the cores is declared a central agent (equivalent to a centralized controller/manager) that is responsible for the DTM decisions for the complete system. However, in case of a dDTM scheme, the behavior of a distributed agent is replicated on each core of the system. Next, we identify a set of functional properties, to be verified for the state-space models, and formally express them using Linear Temporal Logic (LTL) operators available in nuXmv. These LTL properties check the functional correctness of the given DTM for a Many-Core system by verifying its stability w.r.t. threshold avoidance and thermal balancing. For a Many-Core system, stability w.r.t. threshold avoidance is defined as the state when temperature of each core in the system is below a certain threshold. Similarly, stability w.r.t. thermal balancing is the state, where the temperature difference of each core in the system with the average temperature is less then an allowed threshold variation. We have also defined a thermally safe state where both the stability conditions hold for a given Many-Core system. These LTL properties are based on DTM principles, that are common to both cDTM and dDTM in order to provide a common ground for comparative analysis. The LTL properties and the state-space models are then passed to nuXmv for verification. Besides the functional properties, the performance parameters (included in each core module) are also calculated during the verification. The most important performance parameter for a DTM is the number of state transitions it takes to reach a thermally stable state. Other performance properties include parameters like, task load completed, the number of task migrations, the number of tasks stalled, the number of hot spots created and the communication overhead. All these performance parameters are calculated till the system achieves the stable condition.

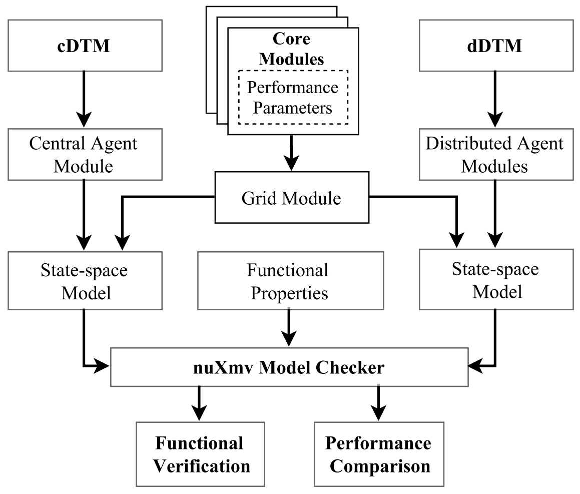

Fig. 1: An Overview of our CAnDy-TM Methodology

III. Analysis of DTM schemes

In this section, we present the formal analysis of two state-of-the-art DTM schemes to show the effectiveness and rigor of our proposed methodology, i.e., CAnDy-TM. To show the generality of our approach, we have selected the DTM schemes from both central and distributed domains:

1) **Multi-Objective DTM - mDTM [6]**. It is a centralized proactive DTM technique that avoids thermal threshold as well as balances the cores temperatures across the chip.

2) **Hot Spots Reduction DTM - hDTM [4].** This technique performs distributed task migration with the primary goal of reducing the number of hot spots in the chip.

More details about these schemes can be found in their respective papers [6], [4].

A. Results and Discussions

The verification of the selected DTM schemes has been done for different grid sizes from 3×3 up to 12×12, using nuXmv 1.0.1, running on a Intel Core i7-6700T Quad-Core server operating at 3.06 GHz with 32 GB of RAM. For modeling the thermal behavior of a core, we have used the following Equation 1 from a real-world Intel Xeon processor [22]:

$$T_i = \Psi_{CA} \times W + T_{LA} \quad (1)$$

where T_i is the core's temperature, Ψ_{CA} is the thermal resistance, W is the power consumption and T_{LA} is the local ambient temperature of the core. The values of variables Ψ_{CA} and T_{LA} given in Equation 1 are taken as 0.655°C/W and 25°C, respectively [22]. For illustrative purposes and for accelerated evaluation, the threshold temperature to invoke task migration is taken as 41.5°C. The values of other parameters used in mDTM and hDTM are taken from their respective implementation details given in [6], [4]. Each task is assigned power units non-deterministically from 1 to 10. For calculating the communication overhead, we have modeled a shared memory system for the Many-Core systems. Therefore, in case of a migration, only a minimal data structure (e.g., pointers to the program and data memory) is transferred from one core to another. We have assumed a 64-bit pointer for our model. In addition to this communication during task migration, each core in a dDTM and cDTM is communicating with its neighbors and the central core, respectively. This communication includes 64 bit values for temperature, core ID and destination core ID for migration. One bit is used to indicate a threshold violation and another bit is used to indicate whether a suitable destination core has been found or not.

1) Functional Verification: The functional verification of mDTM and hDTM is done using the nuXmv's bounded model checking (BMC) support for real numbers with a k-bound value of 15. The verification results show that mDTM ensures thermal safety of the Many-Core system by attaining stability w.r.t. both threshold avoidance and thermal balancing, owing to separate controllers for each purpose. However, hDTM achieves stability only in terms of threshold avoidance and is not stable for thermal balancing and safety. This temperature variance across the chip may cause reliability issues and even result in physically damaging the chip. Therefore, *mDTM is a better choice than hDTM as the foremost objective of a DTM scheme is to achieve thermal stability in a Many-Core system.*

2) Performance Evaluation Results: In addition to thermal stability, we suggest to compare the selected DTM techniques based on the performance properties given by CAnDy-TM. These properties are evaluated till the DTM achieves thermal stability and provide further insights into working of the given DTM schemes. Fig. 2 shows the number of transitions required by each algorithm for different grid sizes to reach thermally stable state with threshold avoidance.

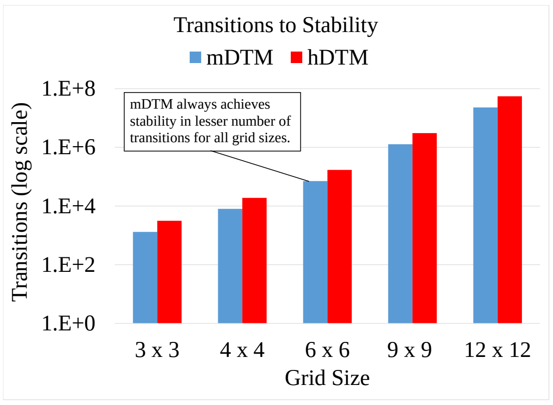

Fig. 2: Number of state transitions to stability

Table I shows the results of other performance properties for different gird sizes till the stability w.r.t. threshold avoidance. We provide the details of some key observations as follows:

Observation 1: The verification of the performance properties shows that the central DTM, i.e., mDTM, outperforms the distributed hDTM technique. For a 12×12 grid size, the number of transitions required by mDTM to achieve threshold stability is 58.4% less than hDTM. It is important to note that this comparison is made w.r.t. the threshold avoidance as hDTM does not meet the thermal safety criterion.

Observation 2: For 12×12 grid size, mDTM performs 40.9% lesser task migrations as compared to hDTM. The fewer number of task migrations means that mDTM achieves stability with lesser overheard in comparison with hDTM. Also, for the same grid size, the performance of mDTM in terms of completion of task load is 62% more than hDTM.

Observation 3: Table II shows the number of state transitions required by mDTM to achieve stability according to different criteria given by CAnDy-TM. For each grid size, the number of transitions required to reach a thermally safe state is always 2.88 and 5.49 times the number of transitions required for achieving thermal balance and threshold stability, respectively.

Observation 4: Fig. 3 shows the average number of communication bits transferred by each core till the stability is achieved. This graph shows that at 3×3 grid size, mDTM requires 4.62 times less number of bits than hDTM. However, at 12×12 grid size, mDTM requires 3.85 times more number of bits than hDTM. This trend shows that *for smaller grid sizes (up to 6×6), the central DTM, i.e., mDTM, performs better than the distributed DTM, i.e., hDTM, in terms of communication overload.*

TABLE I: Verification Results for Performance Properties

Properties		Task Load	Task	Hot	Task
Algorithm	**Grid Size**	**Completed**	**Migration**	**Spots**	**Stalls**
mDTM [6]	3 x 3	56752	558	205	205
	4 x 4	340512	3348	1248	1248
	6 x 6	3064608	30132	11070	11070
	9 x 9	55162944	542376	199260	199260
	12 x 12	992932992	9762768	3586688	3586688
hDTM [4]	3 x 3	35003	945	95	95
	4 x 4	210018	5670	615	615
	6 x 6	1890162	51030	5130	5130
	9 x 9	34022916	918540	92340	92340
	12 x 12	612412488	16533720	1662127	1662127

TABLE II: Transitions to stability criteria for mDTM [6]

Grid Size	Threhold Avoidance	Thermal Balance	Thermal Safety
3×3	1312	2496	7203
4×4	8015	15021	43218
6×6	70848	134784	388962
9×9	1275264	2426112	7001316
12×12	22954752	43670016	126023688

Observation 5: Table III shows that the memory requirement for verification of stability property for mDTM and hDTM. Due to the complexity of mDTM, its verification requires more memory than that for hDTM. For example, for grid sizes 6×6 and up, the verification of mDTM uses full memory resources. However, the verification of hDTM utilizes full memory resources only for the grid size of 12×12. Moreover, the verification of mDTM always takes more time as compared hDTM due to its complexity.

IV. Conclusion

In this paper, we have presented a generic analysis methodology, i.e., CAnDy-TM, for analyzing and comparing central

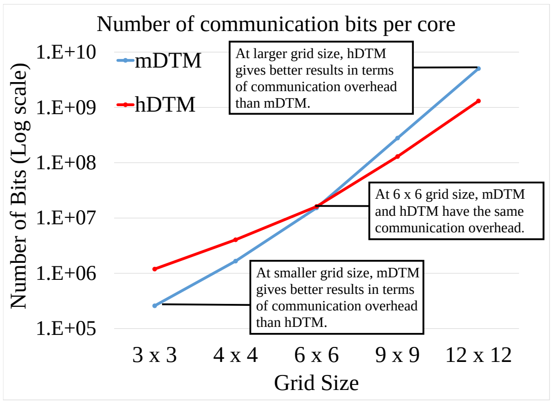

Fig. 3: Comparison of communication overhead per core.

TABLE III: Verification Time and Memory for Threshold Avoidance Stability

Algorithm	Grid Size	Memory (MB)	Verification Time (sec)
mDTM [6]	3 x 3	17845	701
	4 × 4	25186	854
	6 × 6	31289	1595
	9 × 9	31289	8451
	12 × 12	31289	15789
hDTM [4]	3 × 3	8251	245
	4 × 4	12376	342
	6 × 6	19754	987
	9 × 9	28457	4548
	12 × 12	31289	9541

and distributed DTM schemes. Particularly, we have identified a set of formal functional and performance properties that can be formally verified to provide a common comparative framework for different DTM schemes. For illustration purposes, two state-of-the-art DTM schemes have been compared using the nuXmv model checker. Our analysis shows that the central DTM technique, i.e., mDTM [6] achieved a thermally safe state in terms of threshold avoidance and temperature balancing across the chip. On the other hand, the distributed DTM technique, i.e, hDTM [4], achieved stability only in terms of threshold avoidance. Moreover, as compared to hDTM, mDTM performs 62% better in terms of task load completion while reducing the time to threshold stability by 58.4%, and the number of task migrations by 40.9%, respectively. The analysis also reveals that, for smaller grid sizes (up to 6 × 6), mDTM performs better in terms of communication overhead as compared to the hDTM. However, for grid sizes greater than 6 × 6, hDTM schemes perform better than the central mDTM. We believe that the traditional analysis methods, like simulation or emulation, cannot provide a fair comparison between various DTM schemes due to their incompleteness. We have also provided open-source access to the code [21] for CAnDy-TM, as it will benefit in results reproduction and help in designing DTMs with formally verified properties.

Acknowledgement

This work is supported in parts by the German Research Foundation (DFG) as part of the priority program "Dependable Embedded Systems" (SPP 1500 - spp1500.itec.kit.edu) and in parts by the DAAD "Deutsch-Pakistanische Forschungskooperationen" project.

References

[1] W. Huang, K. Skadron, S. Gurumurthi, R. J. Ribando, and M. R. Stan, "Exploring the thermal impact on manycore processor performance," in *Semiconductor Thermal Measurement and Management Symposium*. IEEE, 2010, pp. 191–197.

[2] J. Donald and M. Martonosi, "Techniques for multicore thermal management: Classification and new exploration," in *Computer Architecture*, 2006, pp. 78–88.

[3] R. Mukherjee and S. O. Memik, "Physical aware frequency selection for dynamic thermal management in multi-core systems," in *Computer-aided Design*. ACM, 2006, pp. 547–552.

[4] Z. Liu, X. Huang, S.-D. Tan, H. Wang, and H. Tang, "Distributed task migration for thermal hot spot reduction in many-core microprocessors," in *ASIC*, 2013, pp. 1–4.

[5] M. A. Al Faruque, J. Jahn, T. Ebi, and J. Henkel, "Runtime thermal management using software agents for multi-and many-core architectures," *IEEE Design & Test of Computers*, vol. 27, no. 6, pp. 58–68, 2010.

[6] H. Khdr, T. Ebi, M. Shafique, H. Amrouch, and J. Henkel, "mdtm: multi-objective dynamic thermal management for on-chip systems," in *Design, Automation & Test in Europe*, 2014, pp. 330–336.

[7] M. Kadin, S. Reda, and A. Uht, "Central vs. distributed dynamic thermal management for multi-core processors: Which one is better?" in *Great Lakes Symposium on VLSI*. ACM, 2009, pp. 137–140.

[8] T. Ebi, M. Faruque, and J. Henkel, "Tape: Thermal-aware agent-based power economy for multi/many-core architectures," in *Computer-Aided Design*, 2009, pp. 302–309.

[9] B. Yun, K. G. Shin, and S. Wang, "Predicting thermal behavior for temperature management in time-critical multicore systems," in *Real-Time and Embedded Technology and Applications*, 2013, pp. 185–194.

[10] B. Wojciechowski, K. Berezowski, P. Patronik, and J. Biernat, "Fast and accurate thermal simulation and modelling of workloads of many-core processors," in *Thermal Investigations of ICs & Systems*, 2011, pp. 1–6.

[11] S. Ananthanarayanan, C. Ravishankar, S. Garg, and A. Kennings, "Empower: FPGA Based Emulation of Dynamic Power Management Algorithms for Multi-core Systems on Chip," in *International Symposium on Field Programmable Gate Arrays*, 2012, pp. 266–266.

[12] D. Dunn, "Intel delays Montecito in roadmap shakeup," *EE Times, Manufacturing/Packaging*, 2005.

[13] G. Norman, D. Parker, M. Kwiatkowska, E. Shukla, and R. Gupta, "Using probabilistic model checking for dynamic power management," *Formal Aspects of Computing*, vol. 17, pp. 202–215, 2003.

[14] A. Lungu, P. Bose, D. J. Sorin, S. German, and G. Janssen, "Multicore power management: Ensuring robustness via early-stage formal verification," in *Formal Methods and Models for Codesign*, 2009, pp. 78–87.

[15] S. Iqtedar, O. Hasan, M. Shafique, and J. Henkel, "Formal probabilistic analysis of distributed dynamic thermal management," in *Design, Automation & Test in Europe*, 2015, pp. 1221–1224.

[16] ——, "Probabilistic formal verification methodology for decentralized thermal management in on-chip systems," in *Enabling Technologies: Infrastructure for Collaborative Enterprises*, 2015, pp. 210–215.

[17] E. M. Clarke, Jr., O. Grumberg, and D. A. Peled, *Model Checking*, 1999.

[18] S. A. A. Bukhari, F. K. Lodhi, O. Hasan, M. Shafique, and J. Henkel, "Formal Verification of Distributed Task Migration for Thermal Management in On-chip Multi-core Systems using nuXmv," in *Formal Techniques for Safety-Critical Systems*, ser. Communications in Computer and Information Science, vol. 476, 2015, pp. 32–46.

[19] R. Cavada, A. Cimatti, M. Dorigatti, A. Griggio, A. Mariotti, A. Micheli, S. Mover, M. Roveri, and S. Tonetta, "The nuXmv Symbolic Model Checker," in *Computer Aided Verification*, ser. LNCS. Springer, 2014, vol. 8559, pp. 334–342.

[20] S. A. A. Bukhari, F. K. Lodhi, O. Hasan, M. Shafique, and J. Henkel, "FAMe-TM: Formal analysis methodology for task migration algorithms in many-core systems," *Science of Computer Programming*, vol. 133, Part 2, pp. 154 – 174, 2017.

[21] "Sources for CAnDy-TM," http://save.seecs.nust.edu.pk/projects/candy-tm/, 2017.

[22] Intel, "Intel xeon processor 7400 series thermal/mechanical design guidelines," www.intel.com/Assets/en_US/PDF/designguide/320337.pdf, 2016.