Performance Optimization of Persistent Memory Systems Through Phase-Based Transactional Memory

Rafael Murari[1], João Paulo Carvalho[2], Guido Araujo[2], Alexandro Baldassin[1]

[1]Department of Statistics, Applied Mathematics and Computing –
São Paulo State University (UNESP) – Rio Claro, Brazil

[2]Institute of Computing – University of Campinas (UNICAMP) –
Campinas, Brazil

{rafael.murari,alexandro.baldassin}@unesp.br

{joao.carvalho,guido}@ic.unicamp.br

Abstract. *The emerging persistent memory technologies (PM) are aimed to eliminate the gap between main memory and storage. Nevertheless, its adoption requires measures to guarantee consistency, since crash failures might render the program in an unrecoverable state. In this context, the usage of durable transactions is one of the main investigated approaches to ease the adoption of PM. However, today's implementations are based exclusively on software (SW) or hardware (HW), which might degrade system performance. This paper presents NV-PhTM, a transactional system for PM that delivers the best out of both HW and SW transactions by dynamically changing the execution according to the application's characteristics.*

Resumo. *As emergentes tecnologias de memória persistente (PM) visam eliminar a lacuna existente entre a memória principal e a secundária. No entanto, sua adoção requer medidas para garantia de consistência, visto que possíveis falhas de sistema podem resultar em um estado irrecuperável. Neste contexto, o uso de transações duráveis é uma das abordagens mais investigadas para facilitar a adoção da PM. Todavia, as implementações atuais baseiam-se exclusivamente em software (SW) ou hardware (HW), podendo resultar em degradação de desempenho. Este trabalho apresenta a solução NV-PhTM, um sistema transacional baseado em fases capaz de alterar dinamicamente o modo de execução, SW ou HW, mediante as características apresentadas pelas aplicações.*

1. Introduction

The last few years have witnessed a disruption in the traditional memory hierarchy due to the appearance of novel byte-addressable non-volatile memory technologies, commonly referenced as NVM or PM [Mutlu and Subramanian 2014]. These technologies, characterized by high density and fast access time, are able to expose persistent storage to applications through standard memory instructions (*load* and *store*), avoiding the high overhead involved in intermediate software layers, like drivers and file system. Nonetheless, the adoption of PM is not straightforward, as crash failures might persistently corrupt whole memory regions. In order to address this challenge, the usage of durable transactions has been proposed due to its strong semantics and ease-of-use idiom (popularized by database systems) [Harris et al. 2010].

Early works focused on providing software durable transactions (STM), adding a logging scheme to track all persistent writes. However, this leads to a poor performance when executing short-lived transactions. Therefore, recent works have proposed using hardware transactional support (HTM), available in current Intel and IBM microprocessors, as a way of reducing the high overhead introduced by software transactional systems. Despite its good performance, contemporary HTM systems only provide a *best-effort* implementation, relying on a *fallback* mechanism, usually the acquirement of a Single Global Lock (SGL), to guarantee system progress.

In this context, this paper presents NV-PhTM: a durable phase-based transactional system that executes transactions in phases (HW or SW) selected according to the application's characteristics. Contrary to current systems that resort to serialization, NV-PhTM can smoothly transition the execution to durable software transactions, maintaining a high degree of parallelism. Experimental results with the STAMP benchmark [Minh et al. 2008] show the feasibility of NV-PhTM in detecting and following the best execution mode. This article is divided as follows. Section 2 describes NV-PhTM design. Section 3 presents the experimental evaluation of the new system, comparing it against other state-of-the-art approaches. Finally, Section 4 presents the conclusion.

2. NV-PhTM

Aimed to identify the best execution mode for the distinct transactional specificities, the Non-Volatile Phased Transactional Memory (NV-PhTM) uses the feedback information, from transactions, together with a set of heuristics to guide the transition between distinct transactional implementations (HW and SW). NV-PhTM is built upon PhTM* [Carvalho et al. 2018], a phase-based transactional system developed for volatile memory systems, however certain extensions were proposed: (i) the addition of the durability concept, fulfilling the ACID (Atomicity, Consistency, Isolation and Durability) properties; (ii) new strategies elaborated to guarantee persistency between phase transitions; (iii) design of new transition heuristics that consider the nature of durability strategies and the characteristics of PM.

The architecture designed for NV-PhTM aims to maintain the modularity proposed by PhTM*, which allows the replacement of HW and SW modules by other implementations that provide the same set of operations defined in each transactional API. However, the limitations presented by HTM solutions for PM (e.g., hardware extensions) impeded the hardware modularity, being entrusted to NV-HTM [Castro et al. 2018] the provision of this mode. The modularity of NV-PhTM is restricted to the SW mode, enabling the evaluation of different implementations in order to obtain better performance. The effectiveness of modularization lies in the independence of execution between modes. In this context, the architecture was designed considering the following aspects: (i) HW and SW modes use the same persistent memory regions (log and application state); (ii) before the end of the transition, the system runs a consolidation routine in order to persist the last consistent state produced by current mode; (iii) the effectiveness of the proposed heuristics, aiming at reduce the number of transitions performed by the system.

Persistent memory regions allocated for application state and logs are shared between the distinct implementations (HW and SW), that is, both update these regions but in different phases of the system. Such an organization enables the efficient reuse of memory, reducing the footprint used by the system to ensure persistence. However, certain precautions should be taken during the transition between modes in order to avoid possible race conditions, arising from the different strategies used to guarantee the persistency of the application state. While the NV-HTM delegates this task to a checkpoint scheme, the SW mode, implemented by PSTM [Avni et al. 2015], writes transactional work directly to persistent memory at commit stage. Thus, prior to the transition between phases, it is necessary that the persistent state of the application, produced by the current mode, reflects the last consistent state, ensuring the isolation of each mode.

3. Experimental Evaluation

All experiments were conducted on an Intel Xeon E5-2660 v4 2.0GHz processor with 14 physical cores, each with 2 hardware threads, for a total of 28 concurrent threads. The hyper-threading mode was not used as it tends to decrease performance due to capacity aborts [Carvalho et al. 2018]. The machine is equipped with 64GB of RAM and runs CentOS 7 Linux kernel version 3.10. For space constraints, the results reported in this section, obtained by the mean of 30 executions, are only a subset of the STAMP benchmark, namely Genome and Vacation, which represent the performance behavior of the benchmark. Figure 1 shows the execution time comparison between 3 distinct durable transactional systems: NV-HTM (hardware-only), PSTM (software-only) and NV-PhTM (phase-based).

Genome presents a hardware-friendly behavior due to low abort rate, around 7%, which does not require frequently serialization to guarantee system progress. Thereby, NV-PhTM is able to detect such peculiarity and keep executing transactions in HW mode (NV-HTM). On the other hand, Vacation has a reasonable abort rate, around 16%, most of them as a result of capacity aborts. Thus, the system resort to the fallback mechanism to commit these transactions, leading to a poor performance. When using 6 threads or more, NV-PhTM starts to follow the software implementation (PSTM), avoiding overheads and achieving a high efficiency.

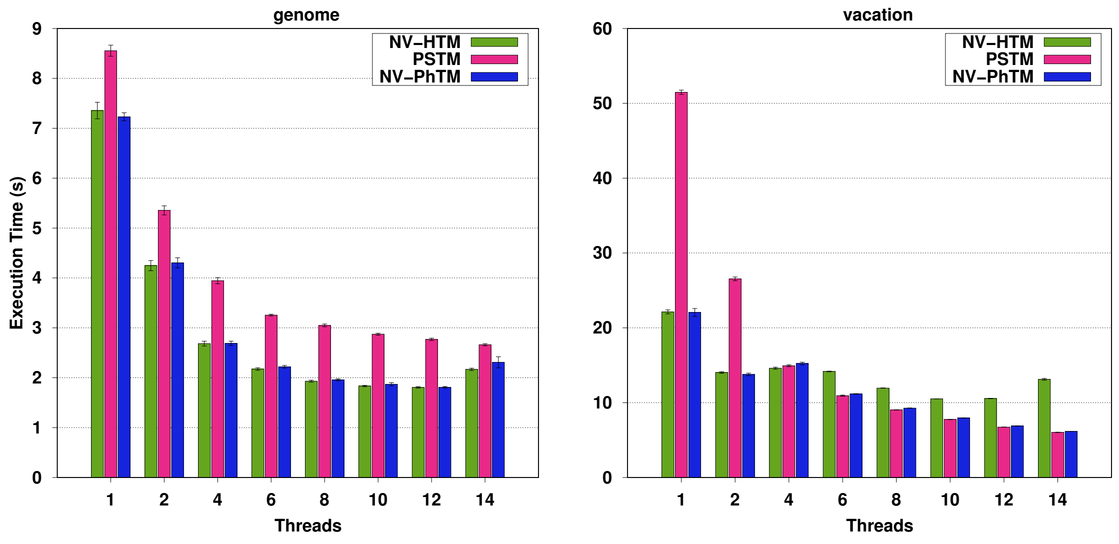

Figure 1. Subset of benchmark STAMP.

The experimental results using the subset of STAMP benchmark show that the extra cost, caused by the addition of the persistent state consolidation routine during phase transitions, is low, reaching a maximum of 0.04% at Genome execution time. Consequently, it does not impact the system performance.

4. Conclusion

As an alternative to existing solutions, this work presents a phase-based transactional system (NV-PhTM), which uses the feedback returned by transactions together with a set of heuristics to guide the transition between different transactional implementations. The main contributions of this work are: (i) the addition of the concept of durability in phase-based systems; (ii) the design of new heuristics that take into account the specificities presented by PM; (iii) the development of strategies to ensure consistency during the transition between the different modes.

The experimental evaluation demonstrated the success of NV-PhTM in detecting and guiding phase transitions with low overhead. When compared to NV-HTM (hardware-only durable transactions) and PSTM (software-only durable transactions), NV-PhTM provided the best overall results due to its nature of following the best performing system.

References

Avni, H., Levy, E., and Mendelson, A. (2015). Hardware transactions in nonvolatile memory. In *Proceedings of the 29th International Symposium on Distributed Computing - Volume 9363*, DISC 2015, pages 617–630, Berlin, Heidelberg. Springer-Verlag.

Carvalho, J. P. D., Araujo, G., and Baldassin, A. (2018). The case for phase-based transactional memory. *IEEE Transactions on Parallel and Distributed Systems*.

Castro, D., Romano, P., and Barreto, J. (2018). Hardware transactional memory meets memory persistency. In *2018 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 368–377, Vancouver, BC, Canada. IEEE.

Harris, T., Larus, J., and Rajwar, R. (2010). *Transactional Memory*. Morgan and Claypool Publishers, 2nd edition.

Minh, C. C., Chung, J., Kozyrakis, C., and Olukotun, K. (2008). Stamp: Stanford transactional applications for multi-processing. In *2008 IEEE International Symposium on Workload Characterization*, pages 35–46, Seattle, WA, USA. IEEE.

Mutlu, O. and Subramanian, L. (2014). Research problems and opportunities in memory systems. *Supercomputing Frontiers and Innovations: an International Journal*, 1(3):19–55.