1 Summary

Transactional Memory: Architectural Support for Lock-Free Data Structures (1993): The authors offer a multiprocessor architecture that accomodates easy-to-use lock-free synchronization, performing atomic updates without the general problems seen with locks/mutual exclusion (priority inversion, convoying, and deadlocks). This architecture is called transactional memory (TM), which is implemented as extensions to multiprocessor cache-coherence protocols. Transactional memory is shown to perform as well as, if not better than, conventional lock-based techniques for atomic updates of the time of the paper, producing higher throughput than the other hardware- and software-based lock-based mechanisms at all levels of concurrency in most cases for simple benchmarks. Since TM does not use locks, it requires fewer accesses to shared memory and therefore accomplishes work in fewer cycles.

2 Strengths

- The experiment compares TM to a variety of other competitive lock mechnisms, benchmarking against both hardware- and software-based mechanisms for mutual exclusion.
- The paper clearly characterizes the benchmarks they created (Sec. 5), highlighting key features that their architecture hoped to target and perform well with in real applications. They discuss reasonable explanations for certain performance difference such as cost of contention with lock mechanisms, while also recognizing the limitations of their implementation and methodology, namely how small their critical sections are and them accommodating short transactions.

3 Weaknesses

- The paper only measures performance outcomes with elapsed time between the different atomic updating techniques, rather than also reporting other metrics, such as IPC or power consumption.
- The experiment did not test the TM on a more varied set of applications with different kinds of memory access patterns. The paper only focuses on three simple benchmarks (counting, DLL, and producer/consumer).

4 Rating: 4

5 Comments

Herlihy and Moss' work has held up since they first published on TM in 1993, mainly in academic research. Different abstractions and/or implementations of TM have been studied and published even as recently as within the last few years. Harris, Larus and Rajwar published on the topic in the 2010 Synthesis Lectures in Computer Architecture [2], where Herlihy's recent works at the time were still referenced. TM is found to be used in some software and hardware applications. `gcc` offers the transactional memory library with the `-fgnu-tm` flag [3]. `GHC` offers a similar abstraction with the STM [4]. ARM offers hardware transaction memory extension (TME) [5]. And IBM had TM implemented on certain systems, such as the Blue Gene/Q (supercomputer) and zEC12 (mainframe), but each were discontinued in 2015 and 2016, respectively. More systems are listed in the Syntesis Lecture, but it appears lack of proliferation in the market is due to difficulties in integrating existing TM mechanisms with other systems. Otherwise, TM remains an active area of research because it holds promise as a solution for parallel programming, and it is likely industries will try implementing more reliable TM for their systems. It was not necessarily simple finding out why systems with TM were few or decisively why they get discontinued by major companies, but some likely reasons may be maintainability and cost, functionality and correctness of computation, and large abort rates for larger applications, the later of which was an open problem for the paper given that TM was a new architecture at the time. Given how large software applications have grown even in more recent years, the more important problems to address in TM may be reliabilty of such systems and reducing the abort rates.

References

[1] Maurice Herlihy and J. Eliot B. Moss. 1993. "Transactional memory: architectural support for lock-free data structures. In Proceedings of the 20th annual international symposium on computer architecture (ISCA '93). ACM, New York, NY, USA, 289-300. http://dx.doi.org/10.1145/165123.165164

[2] Tim Harris, James Larus, and Ravi Rajwar. 2010. Transactional Memory, 2nd Edition. Synthesis Lectures on Computer Architecture. Springer Cham.

[3] The GNU Transactional Memory Library. Free Software Foundation, Inc. 2015.

[4] Software Transactional Memory (STM). Haskell Wiki. 2022.

[5] Overview of Arm Transactional Memory Extension (TME). Armv9-A Documentation.

[6] Transactional Memory. IBM. 2021.