A Comprehensive Scheme for Contention Management in Hardware Transactional Memory

Xiaoqun Wang*, Zhenzhou Ji, Chen Fu, and Mingzeng Hu
Harbin Institute of Technology, Harbin 150001, China
wxiaoqun@gmail.com

Abstract. Transactional Memory (TM) is one kind of approach to maximize parallel performance for multicore systems. There are conflicts when two or more parallel transactions access the same location and at least one access is a write. Contention management (CM) refers to the mechanisms used to guarantee forward—to avoid performance pathology, and to promote throughput. In this paper, we introduce a new CM police. We remitted six of seven performance pathologies summered by Bobba. Our result shows high performance for large transactions, while get moderate improvement or little slowdown for small transactions. The performance of the systems used this policies combined with other policy are steady.

Keywords: hardware transactional memory, contention management, parallel programming, multicore processors.

1 Introduction

Transactional memory systems (TM) [1] are expected to be a technique for parallel programming for general-purpose computing. Especially for hardware transactional memory (HTM) systems, the programmers need not worry about the correctness of a multithreaded application. Generally, transactional memory systems can exceed the performance of lock based systems which serialize all the transactions when they share data. When the executing workload lacks sufficient parallelism, lock based systems do not hurt the performance since they were pessimistically serializing every transaction. For a transactional memory system, excessive transactions can actually result in considerate conflicts. A conflict is defined that when two or more concurrent transactions access the same location and at least one access is a write.TM systems resolve them by CM policies.

* This work is supported by the National High Technology Development 863 Program of China under grant #2006AA04A103. In the meanwhile, this work is also supported by the National Science Foundation of China under grant #60503015 and the ShanDong Province Science and Technology Development Foundation under grant #2007GG10001020.
The concept of “contention management” is introduced by Herlihy et al. in the context of software transactional memory (STM) up to date, many CM policies are proposed for resolve conflicts. Scherer et al. studies a set of arbitration heuristics on the STM framework. They firstly use transactions gather information such as priority, read/write set size, number of and aborts. Then they it them decide which transactions will be aborted. However they did not evaluate conflict detection time as an important design choice. In most HTM systems,

The tradeoff of version management and conflict detection is the important research focus. Contention management is always ignored. Bobba et al. firstly studied the performance pathologies due to specific conflict detection, contention management, and version management policies in HTMs. They proposed enhanced HTMs to remit specific pathologies. Furthermore, they analyzed specific points in the design space, making it difficult to choose a universal policy which can be effective for TM applications with different kinds of characteristics. TxLinux exploits HTM and integrates transactions with the operating system scheduler. They focus on achieving synchronization in the kernel for future processors with TM hardware support. Their goals and approach is different than our work.

In this paper, we introduce a new CM police. Our study across a wide set of applications makes the following contributions:

1. We remitted six of seven performance pathologies summered by Bobba.
2. Our result shows that our techniques can effectively remit six performance pathologies and the tendency of deadlock, minimize the amount of waste time and consequently, decrease execution time than previous approaches.
3. Our result shows that the performance of the systems used this policy combined with other policies are steady.

2 TMProf

Yen proposes TMProf which is an online lightweight profiling framework to diagnose and analysis the performance a TM system. TMProf implements a set of hardware performance counters to track the execution cycles of the eight transactional related events in each processor core. They are defined as follows:

Useful transaction: Cycles executed by successful transactions that commit.
Wasted transactions: Cycles executed by lose transactions that eventually abort.
Non-transaction: Cycles spent by non-transaction instructions.
Commit: Cycles spent by committing operations.
Stall: Cycles spent by waiting for conflict to resolve.
Abort: Cycles spent by processing aborts.
Backoff: Cycles spent in backoff.
Barrier: Cycles spent in barrier.