Towards Distributed Software Transactional Memory Systems*

Paolo Romano
INESC-ID/IST
romanop@gsd.inesc-id.pt

Nuno Carvalho
INESC-ID/IST
nonius@gsd.inesc-id.pt

Luís Rodrigues
INESC-ID/IST
ler@ist.utl.pt

Abstract

The recent architectural trend that has lead to the widespread adoption of multi-core CPUs has fostered a huge research interest towards Software Transactional Memory (STM). As STMs are starting to face the high availability and scalability requirements of real-world production environments, it is natural to foresee the need for replication solutions specifically tailored for STMs. Since databases and STMs share the same key abstraction of atomic transaction, it is natural to wonder whether the mechanisms originally designed for database replication could be successfully and seamlessly exploited also to support replication of STM systems.

This paper seeks an answer to this question, highlighting some critical performance issues related to the application of state of the art database replication techniques in the context of STM systems, and presenting some of our research directions aimed at designing and implementing high performance replication strategies able to fit the unique requirements of STMs.

1 Introduction

Software Transactional Memory (STM) systems have garnered considerable interest of late due to the recent architectural trend that has lead to the pervasive adoption of multi-core CPUs. STMs represent an attractive solution to spare programmers from the pitfalls of conventional explicit lock-based thread synchronization, leveraging on proven concurrency-control concepts used for decades by the database community to simplify the mainstream

*This paper was partially supported by the Pastramy (PTDC/EIA/72405/2006) project.

parallel programming [2]. When using STMs the programmers are simply required to specify which operations on shared data structures are to be executed within the scope of an atomic and isolated transaction. The task of ensuring the consistent execution of the transactions is delegated to the STM, which transparently takes care of regulating the concurrent access to shared data, by automatically aborting unserializable transactions, avoiding deadlocks and priority inversions.

When STM systems are used in the core of enterprise systems, they are faced with the high availability and scalability requirements of production environments [8]. A relevant example is the FénixEDU system, which is a web application that relies on a STM-based solution in order to ensure the consistency of an in memory middle-tier object cache. The current version of the FénixEDU system is facing scalability and dependability challenges, as it has to process between 1,000,000 and 4,500,000 transactions per day for a population of 12000 students, 900 faculty and 800 administrative members in the Instituto Superior Técnico (IST) campus[1]. As replication techniques have been successfully used since decades to enhance the availability of computing systems, it is rather natural to foresee the emergence of replication solutions specifically tailored for STM systems. Also, since STM and Database management Systems (DBMS) share the notion of transaction, it might appear that the state of the art database replication schemes [21, 20, 9, 3] represent natural candidates, among the plethora of replication solutions presented in literature, to support STM replication.

This paper aims at uncovering the pitfalls related to the adoption of conventional database replication techniques in the context of STM systems. First, in Section 2, we recall some key techniques at the basis of the most popular modern database replication techniques. In the Section 3 we contrast, from a replication oriented perspective, the workload characteristics of two standard benchmarks for STM and DBMS. Our analysis allows us to derive insights on several critical issues which make the state of the art solutions based on database replication strongly under-performing when adopted in the context of STM-based systems. At the light of such analysis, in Section 4, we present promising research directions we are currently pursuing in order to develop high performance replication strategies able to fit the unique characteristics of the STM.

[1] `http://www.ist.utl.pt`

2 Database Replication

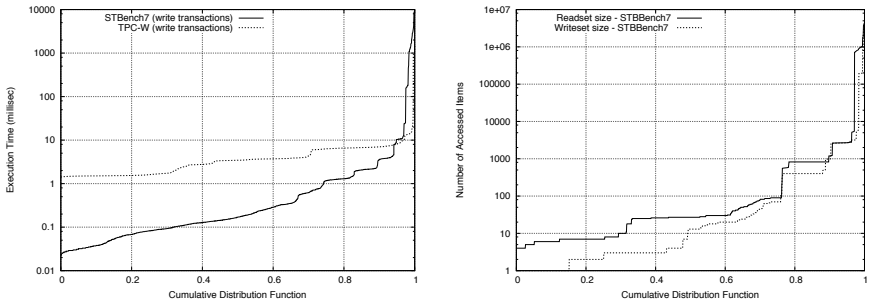
Most recent database replication schemes perform reads on one replica and writes on all (available) replicas so to maximize performance of (common) read-intensive applications. Further, the fulcrum of modern database replication schemes [21, 20, 9, 3, 17, 10] is the reliance on an Atomic Broadcast [11, 12] (AB) primitive, typically provided by some Group Communication System (GCS) [19, 4]. AB plays a key role to enforce, in a non-blocking manner, a global transaction serialization order without incurring in the scalability problems affecting classical eager replication mechanisms [13].

The abundant AB-based database replication literature can be classified in two main categories depending on whether the AB primitive is used to multicast (*i*) the initial transactional "request", rather than (*ii*) the sets of items accessed by the transaction that enters the commit phase. In the former case, the approach is often referred to as state machine replication since, just like in the original state machine approach [23], a (transactional) request is serialized in the same order at all the replicas *prior* to its execution [18]. In the latter solutions, which are typically referred to as "certification-based", transactions are validated *a posteriori* of their execution on the basis of their readsets and writesets. The certification based approaches can be further classified into voting and non-voting schemes [17, 22], where voting schemes, unlike non-voting ones, need to atomic broadcast only the writeset (which is typically much smaller than the readset in classical database workloads), but on the other hand incur the overhead of an additional uniform broadcast [16] along the critical path of the commit phase.

3 Replication on DBMS and STM Systems

In this section we aim at highlighting some critical issues related to the application of state of the art database replication techniques in the context of STM based systems. Our discussion is supported by a quantitative comparative analysis of the workloads of a recent STM benchmark, namely STMBench7 [14], and of a popular benchmark for web-based transactional applications, namely TPC-W [24].

In Figure 1(a) we compare the execution latency of sequentially submitted transactions for STMBench7 and TPC-W when executing the two benchmarks on top of a 4 Xeon CPUs machine using Linux 2.6.8-24.18 and equipped with 2 SCSI disks in RAID-0 configuration and 4GB of main memory. We use JVSTM [7] as the STM for STMBench7, and PosgreSQL 8.1 as



(a) Transaction execution time: STMBench7 vs TPC-W.

(b) Readset/writeset size of STMBench7's transactions

Figure 1: Cumulative Distribution Function (CDF).

the relational database underlying TPC-W. This choice is motivated by that both JSVTM and PostgreSQL have a similar approach to concurrency control, both of them relying on a multi-versioning scheme which allows read-only transactions to be executed without ever being blocked or aborted. Also, in order to fairly compare the performance of the two systems, we configured the benchmarks so to generate a very similar percentage of write transactions: specifically we evaluated the TPC-W *Ordering Mix* and the STMBench7 *Read-Write Workload Type*, whose percentage of write transactions is around the 50%.

By the plots we can draw the following considerations. First, the transaction execution time for the STMBench7 is at least one order of magnitude smaller than for TPC-W for around the 60% of transactions. This is essentially due to that, unlike conventional DBMS transactions, STM transactions only access in memory data items, hence not incurring the latencies proper of disk accesses. Also, in STMs the overhead associated with SQL parsing and plan optimization are absent, further contributing to shortening the transaction lifetime.

The direct consequence of such a striking reduction of the transaction lifetime in STMs with respect to DBMSs is that, in a replicated STM system, the relative overhead of the atomic broadcast based synchronization schemes would be correspondingly amplified with respect to the scenario of conventional database replication.

Another important feature characterizing the STM benchmark is the high heterogeneity of its workload. In fact, the transaction lifetime in STMBench7, as clearly highlighted by Figure 1(a), spans over an impressively

wide range, with around the 30% of transactions executing for less than 100 micro-seconds, and about the 5% of transactions taking from hundreds of milli-seconds up to several seconds. This is also reflected by the high heterogeneity of the sizes of the transaction readsets and writesets, see Figure 1(b), which range from a just a few items up to around 10 millions. Indeed, the presence of highly diversified components in the workload of STM applications, such as in STMBench7, severely challenges the state of the art on database replication solutions, which are designed to provide optimal performance under much more restricted workloads. Another interesting consideration that can be drawn by observing the writesets' CDF in Figure 1(b), is that the writesets' and readsets' cardinality of STM transactions is quite similar, especially when considering long-running transactions (representing nearly the 5% of the STMBench7 workload) which reads and writes from tens of thousands to millions of data items. This strongly contrasts with classical database workloads, in which transactions are rather characterized by small writesets and often very large readsets, and for which existing database replication solutions have been optimized.

4 Current Research Directions

In order to tackle the issues highlighted in the previous section, we are currently pursuing several orthogonal, yet complementary, research directions which address both the theoretical and engineering aspects of scalable and resilient STM implementation:

Speculative transaction execution The average latency of AB run is, even for very small messages, on the order of at least a few milliseconds in typical data-center environments, see, e.g., [15, 1, 5]. By contrasting this performance data with the completion time of (not replicated) STM transactions (see Figure 1(a)), we can argue that nearly the 50% of STM transactions complete in around 1/10th of the time required to perform an AB. This implies that it is highly likely that transactions complete and stall (relatively) long before the AB is concluded, which could lead to severe under-utilization of the available computing resource.

Given the above considerations, we are currently pursuing the idea to speculatively explore multiple alternative transaction serialization orders (rather than just the one suggested by the spontaneous order delivery as suggested in, e.g. [18]) so to maximize the utilization of any CPU core waiting idle for the termination of an AB's run. Interestingly, this approach seems,

in principle, applicable to both certification-based replication schemes [17] and classical state machine replication approaches [23].

The main challenge here is related to that the number of possible serialization orders over a set composed of n elements is $n!$, which drastically reduces the probability to blindly select the correct final serialization order as the number of messages to be ordered grows. This issue raises the need for ingenious heuristics that are able to effectively maximize the probability to drive the speculative exploration of the serialization orders towards useful trajectories.

Space efficient transaction readset's encoding via Bloom Filters The efficiency of AB is known to be strongly affected by the size of the exchanged messages [15, 1, 5]. A straightforward way to improve the performance of existing AB-based replication schemes is therefore to introduce mechanisms capable of effectively reducing the size of the messages multicast trough the AB primitive. To pursue this goal, we are studying how to exploit Bloom filters[2] [6] to encode in a space-efficient way the transactions' read-/write-sets that are exchanged by certification based protocols.

The algorithmic and engineering challenge here is how to effectively exploit the message compression achievable trough Bloom filters, while preserving the system's consistency and minimizing the performance drawbacks due to the occurrences of false positives.

Adaptive replication strategies We argue that no single universal replication scheme exists that is able to effectively cope with the high heterogeneity characterizing STM workloads. Therefore we advocate the need for developing self-adapting STM replication schemes, able to timely and automatically identify the optimal replication strategy for each incoming transaction on the basis of the (estimated) size of its readset and writeset, as well as of its conflict probability. It is in fact possible to show that replication schemes, such as the certification based solution [3] (in both its voting and non-voting variants [22]) and the state machine scheme [23], are designed to provide optimal performances in distinct (i.e. not overlapping) regions of the space identified by the Cartesian product of the above parameters.

Realizing such a polymorphic replication strategy requires facing two main challenges: on one hand, ensuring the consistent interaction of different

[2]Bloom filters are simple randomized data structures for supporting set membership queries, which allow trading off the space compression factor for the possibility of incurring in false positives.

replication algorithms and, on the other hand, engineering lightweight and timely mechanisms to automatically identify the characteristics of incoming transactions and the corresponding preferable replication scheme.

5 Conclusions

In this paper we discussed several characterizing features of STM systems' workloads and highlighted some crucial issues related to the application of state of the art database replication solutions in the context of STM systems. We then presented some of the research directions we are currently pursuing to develop high performance, dependable replicated STM systems.

In the LADIS workshop we plan to present some preliminary results of our current research activities based on experimentation with reference STM benchmarks.

Our final aim is, however, to assess the effectiveness of the proposed techniques on the FénixEDU system, which is, to the best of our knowledge, the first STM system in the world to be used in a (large scale) production environment.

References

[1] An evaluation of the amoeba group communication system. In *Proce. of the 16th International Conference on Distributed Computing Systems*, page 436, Washington, DC, USA, 1996. IEEE Computer Society.

[2] A.-R. Adl-Tabatabai, C. Kozyrakis, and B. Saha. Unlocking concurrency. *ACM Queue*, 4(10):24–33, 2007.

[3] D. Agrawal, G. Alonso, A. E. Abbadi, and I. Stanoi. Exploiting atomic broadcast in replicated databases (extended abstract). In *Proc. of the Third International Euro-Par Conference on Parallel Processing*, pages 496–503, London, UK, 1997. Springer-Verlag.

[4] Y. Amir, C. Danilov, and J. Stanton. A low latency, loss tolerant architecture and protocol for wide area group communication. In *Proceedings of the International Conference on Dependable Systems and Networks (DSN)*, June 2000.

[5] R. Baldoni, E. Barbi, S. Cimmino, C. Marchetti, P. Papa, and L. Querzoni. A Practical Comparison between the TAO Real-Time Event Service and the Maestro/Ensemble Group Communication System. In *In*

proceedings of Distributed Objects and Applications (DOA) 2004, Larnaca, Cyprus, 10 2004.

[6] B. H. Bloom. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM*, 13(7):422–426, 1970.

[7] J. Cachopo and A. Rito-Silva. Combining software transactional memory with a domain modeling language to simplify web application development. In *6th International Conference on Web Engineering*, pages 297–304, July 2006.

[8] N. Carvalho, J. Cachopo, L. Rodrigues, and R. S. A. Versioned transactional shared memory for the fenixedu web application. In *Proc. of the Second Workshop on DependableDistributed Data Management (in conjunction with Eurosys 2008)*, Glasgow, Scotland, Mar. 2008. ACM.

[9] E. Cecchet, J. Marguerite, and W. Zwaenepole. C-jdbc: flexible database clustering middleware. In *Proc. of the USENIX Annual Technical Conference*, pages 26–26, Berkeley, CA, USA, 2004. USENIX Association.

[10] A. J. Correia, J. O. Pereira, L. Rodrigues, N. M. R. Carvalho, R. Vilaça, R. Oliveira, and S. Guedes. Gorda: An open architecture for database replication. In *Proc. of the 6th International Symposium on Network Computing and Applications*, pages 287–290, Boston, MA, USA, july 2007. IEEE, IEEE.

[11] D. Powell (ed.). *Special Issue on Group Communication*, volume 39. ACM, 1996.

[12] X. Defago, A. Schiper, and P. Urban. Total order broadcast and multicast algorithms: Taxonomy and survey. *ACM Computing Surveys*, 36(4):372–421, 2004.

[13] J. Gray, P. Helland, P. O'Neil, and D. Shasha. The dangers of replication and a solution. In *Proc. of the 25th Conference on the Management of Data (SIGMOD)*, pages 173–182. ACM, 1996.

[14] R. Guerraoui, M. Kapalka, and J. Vitek. Stmbench7: a benchmark for software transactional memory. *SIGOPS Oper. Syst. Rev.*, 41(3):315–324, 2007.

[15] R. Guerraoui, R. R. Levy, B. Pochon, and V. Quema. High throughput total order broadcast for cluster environments. In *Proc. of the International Conference on Dependable Systems and Networks*, pages 549–557, Washington, DC, USA, 2006. IEEE Computer Society.

[16] R. Guerraoui and L. Rodrigues. *Introduction to Reliable Distributed Programming*. Springer, 2006.

[17] B. Kemme and G. Alonso. A suite of database replication protocols based on group communication primitives. In *Proc. of the The 18th International Conference on Distributed Computing Systems*, page 156, Washington, DC, USA, 1998. IEEE Computer Society.

[18] B. Kemme, F. Pedone, G. Alonso, and A. Schiper. Processing transactions over optimistic atomic broadcast protocols. In *Proc. of the 19th IEEE International Conference on Distributed Computing Systems*, page 424, Washington, DC, USA, 1999. IEEE Computer Society.

[19] H. Miranda, A. Pinto, and L. Rodrigues. Appia, a flexible protocol kernel supporting multiple coordinated channels. In *Proceedings of the 21st International Conference on Distributed Computing Systems*, pages 707–710, Phoenix, Arizona, Apr. 2001. IEEE.

[20] M. Patino-Martínez, R. Jiménez-Peris, B. Kemme, and G. Alonso. Scalable replication in database clusters. In *Proc. of the 14th International Conference on Distributed Computing*, pages 315–329, London, UK, 2000. Springer-Verlag.

[21] F. Pedone, R. Guerraoui, and A. Schiper. The database state machine approach. *Distributed and Parallel Databases*, 14(1):71–98, 2003.

[22] L. Rodrigues, H. Miranda, R. Almeida, a. M. Jo and P. Vicente. The globdata fault-tolerant replicated distributed object database. In *Proc. of the First EurAsian Conference on Information and Communication Technology*, pages 426–433, London, UK, 2002. Springer-Verlag.

[23] F. B. Schneider. *Replication management using the state-machine approach*. ACM Press/Addison-Wesley Publishing Co., 1993.

[24] Transaction Processing Performance Council. *TPC Benchmark^{TM} W, Standard Specification, Version 1.8*. Transaction Processing Perfomance Council, 2002.