Genuine replication, opacity and wait-free read transactions: can a STM get them all?

Sebastiano Peluso[1,2], Paolo Romano[2], and Francesco Quaglia[1]

[1] Sapienza University, Rome, Italy
[2] IST/INESC-ID, Lisbon, Portugal

Distributed Transactional Memory (DTM) is an attractive programming paradigm that aims at combining the simplicity of Transactional Memory (TM) with the scalability and failure resiliency achievable by exploiting the resource redundancy of distributed platforms.

In these in-memory distributed platforms replication plays a role of paramount importance both for fault-tolerance and performance reasons. The advantages of replication from the fault-tolerance's perspective are quite obvious. From the performance's standpoint, however, replication can become a double-sided weapon.

By maintaining copies of the same datum across multiple nodes, in fact, the efficiency of a DTM platform can be enhanced by allowing more than one node to access that datum locally (sparing the costs of remote RPCs). On the other hand, replication has an inherent cost as it demands additional, costly operations to enforce synchronization among replicas. Hence, if the costs of replication are not outweighed by the benefits it brings, replication may ultimately hinder performance rather than boosting it.

In this talk we focus on the problem of designing efficient replication protocol for large scale DTM platforms. We argue that efficient replication protocols for large scale DTM systems should follow these three key design principles:

- *partial replication*: a key ingredient of any scalable replication protocol is to limit (upper bound) the number of nodes that have to be involved in the processing of a transaction. This implies embracing partial replication schemes, which replicate data across a (typically rather small) number of different nodes.
- *genuineness:* partial replication alone is not sufficient to ensure high scalability. Another fundamental property of a highly scalable partial replication protocol is genuineness [9], which guarantees that the only nodes that are involved in the processing of a transaction are those that maintain (replicas of) data accessed by that transaction. Note that genuineness rules out replication schemes that rely on centralized (and therefore inherently non-scalable) components.
- *wait-free read-only transactions:* read-only transactions dominate most real workloads, and their efficiency should be seen as a key requirement of a DTM

platform. In this sense, a highly desirably property for read-only transactions is wait-freedom [10]. In fact, given that read-only transactions do not write to data items, it seems plausible, that they should (eventually) terminate successfully, regardless of concurrent transactions [2]. Further, in practice, read only transactions are often used to support long running computations. Wait-freedom is particularly important for this type of transactions, as weaker progress guarantees are likely to lead to their starvation.

The literature on replication protocols for transactional systems is extremely rich, and represents an obvious source of inspirations also for DTM systems. On the other hand, TM systems have unique requirements that are not satisfactorily addressed by existing solution [14, 7]. One of these is related to consistency. Unlike conventional database systems, in fact, (D)TM platforms are non-sandboxed environments, and are, as such, more susceptible to the idiosyncrasies of concurrency. This is the key motivation at the basis of the *opacity* [8] consistency criterion, which, roughly speaking, extends the classic serializability criterion [3] in a twofold direction: guaranteeing that also live transactions (and not only committed ones) access a consistent snapshot, and preserving real-time ordering.

Interestingly, despite the abundance of existing replication protocols for transactional systems, we are not aware of any genuine partial replication scheme that guarantees wait-free read-only transactions and opacity. Existing replication protocols, in fact, either embrace weaker consistency models (e.g. eventual consistency [6], repeatable read [15] or Update Serializability [12]), or adopt full replication schemes [11, 4, 5], or do not guarantee wait-freedom for read-only transactions [16].

To the best of our knowledge, the solution that gets closer to enforcing these three desirable properties is GMU [12]. GMU implements a genuine partial replication protocol and relies on a multi-version scheme that ensure that read-only transactions are never aborted (hence achieving wait-freedom). However, GMU ensures Extended Update Serializability [1], a consistency criterion that is strictly weaker than opacity.

Hence we raise the question of whether it is possible to design partial replication protocols that ensure genuineness, opacity and wait-free read-only transactions. We do not have a conclusive answer to this question yet, and we hope that this talk will trigger the interest of the community on this challenging theoretical problem that has very interesting pragmatical motivations.

However, we make an important step towards filling this gap, by presenting GMES a novel genuine partial replication scheme that ensures wait-free read-only transactions and ensures a consistency criterion that we name Extended One Copy Serializability (E1S). E1S provides 1-Copy Serializability (1S) both for committed and executing transactions, resulting strictly stronger than EUS. This guarantees that, just like in opacity, live transactions (even those that may eventually be aborted) always observe a consistent (i.e., equivalent to the one generated by a serial history) state. However, E1S does not require that real-time order is preserved, and is consequently weaker than opacity. Figure 1 depicts the relationships among these consistency criteria. The diagram reports also

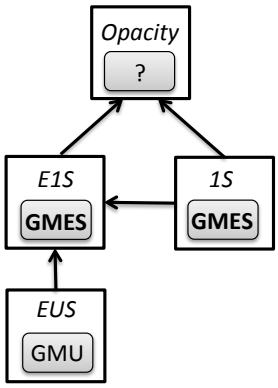

Fig. 1: Relationships among 1S, E1S, EUS and Opacity consistency criteria. We indicate also the name of genuine partial replication protocols that guarantee these criteria and wait-freedom for read-only transactions.

the name of genuine partial protocols that guarantee these consistency criteria and ensure wait-freedom for read-only transactions. As also highlighted in this diagram, GMES, by guaranteeing E1S, obviously ensures also 1S, filling, to the best of knowledge, another gap in the design space of transactional replication protocols. Indeed, we are not aware of any genuine replication protocol that guarantees this consistency criteria and wait-freedom for read-only transactions.

GMES guarantees wait-freedom for read-only transactions via a highly scalable distributed multiversioning scheme and a non-blocking commit protocol layered on top of a genuine total order multicast service [13]. Genuineness is achieved by avoiding to rely on centralized components, such as a shared logical clock [17]. Conversely, snapshot visibility rules are determined by an innovative algorithm that uses distributed, independent logical clocks (one per node) to track data dependencies among transactions. These clocks are advanced in a "genuine" fashion, namely a clock on node n is read/updated only by transactions that access data maintained by n. Dependencies among transactions are tracked, both at commit time and upon the execution of read operations, by synchronizing the advancement of the clocks that maintain the data on which the conflicts are materialized.

References

1. A. Adya. Weak Consistency: A Generalized Theory and Optimistic Implementations for Distributed Transactions. Technical report, PhD Thesis, Massachusetts Institute of Technology, 1999.
2. H. Attiya, E. Hillel, and A. Milani. Inherent limitations on disjoint-access parallel implementations of transactional memory. In *Proceedings of the twenty-first annual*

symposium on Parallelism in algorithms and architectures, SPAA '09, pages 69–78, New York, NY, USA, 2009. ACM.

3. P. A. Bernstein, V. Hadzilacos, and N. Goodman. *Concurrency Control and Recovery in Database Systems*. Addison-Wesley, 1987.
4. N. Carvalho, P. Romano, and L. Rodrigues. Asynchronous lease-based replication of software transactional memory. In *Proceedings of the ACM/IFIP/USENIX 11th International Conference on Middleware*, Middleware '10, pages 376–396, Berlin, Heidelberg, 2010. Springer-Verlag.
5. M. Couceiro, P. Romano, N. Carvalho, and L. Rodrigues. D^{2}STM: Dependable Distributed Software Transactional Memory. In *Proc. of PRDC*. IEEE Computer Society Press, 2009.
6. G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels. Dynamo: amazon's highly available key-value store. In *Proc. of Symposium on Operating Systems Principles*, SOSP '07, pages 205–220. ACM, 2007.
7. P. Felber, C. Fetzer, R. Guerraoui, and T. Harris. Transactions are back—but are they the same? *SIGACT News*, 39(1):48–58, Mar. 2008.
8. R. Guerraoui and M. Kapalka. On the Correctness of Transactional Memory. In *Proc. of ACM Symposium on Principles and Practice of Parallel Programming*, PPoPP '08, pages 175–184. ACM, 2008.
9. R. Guerraoui and A. Schiper. Genuine atomic multicast in asynchronous distributed systems. *Theor. Comput. Sci.*, 254(1-2):297–316, Mar. 2001.
10. M. P. Herlihy. Impossibility and universality results for wait-free synchronization. In *Proceedings of the seventh annual ACM Symposium on Principles of distributed computing*, PODC '88, pages 276–290, New York, NY, USA, 1988. ACM.
11. F. Pedone, R. Guerraoui, and A. Schiper. The Database State Machine Approach. *Distributed and Parallel Databases*, 14(1):71–98, 2003.
12. S. Peluso, P. Ruivo, P. Romano, F. Quaglia, and L. Rodrigues. When Scalability Meets Consistency: Genuine Multiversion Update-Serializable Partial Data Replication. In *Proc. of International Conference on Distributed Systems*, 2012.
13. L. Rodrigues, R. Guerraoui, and A. Schiper. Scalable atomic multicast. In *ICCCN*, pages 840–847, 1998.
14. P. Romano, N. Carvalho, and L. Rodrigues. Towards distributed software transactional memory systems. In *Proc. of the Workshop on Large-Scale Distributed Systems and Middleware*, Sept. 2008.
15. P. Ruivo, M. Couceiro, P. Romano, and L. Rodrigues. Exploiting total order multicast in weakly consistent transactional caches. In *PRDC'11: Proceedings of the 17th Pacific Rim International Symposium on Dependable Computing*, Pasadena, California, USA, Dec. 2011.
16. N. Schiper, P. Sutra, and F. Pedone. P-Store: Genuine Partial Replication in Wide Area Networks. In *Proc. of IEEE Symposium on Reliable Distributed Systems*, pages 214–224, 2010.
17. D. Serrano, M. Patio-martinez, R. Jimenez-peris, and B. Kemme. Boosting database replication scalability through partial replication and 1-copy-snapshotisolation. In *Proceedings of the Pacific Rim International Symposium on Dependable Computing*, 2007.