Journal of Computer and System Sciences 80 (2014) 257–276

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

On speculative replication of transactional systems

Paolo Romano [a,*], Roberto Palmieri [b], Francesco Quaglia [b], Nuno Carvalho [a], Luis Rodrigues [a]

[a] INESC-ID, Instituto Superior Técnico, Universidade Técnica de Lisboa, Portugal
[b] Sapienza Università di Roma, Italy

ARTICLE INFO

Article history:
Received 17 December 2010
Received in revised form 28 November 2012
Accepted 22 July 2013
Available online 6 August 2013

Keywords:
Fault tolerance
Distributed protocols
Software transactional memories

ABSTRACT

In this paper we investigate, from a theoretical perspective, the problem of how to build speculative replication protocols for transactional systems layered on top of an Optimistic Atomic Broadcast (OAB) service. The OAB service provides an early, possibly erroneous, guess on transaction's final serialization order. This can be exploited to speculatively execute transactions in parallel with the algorithm used to determine their final total delivery (and serialization) order. To maximize the chances of guessing their final serialization order, transactions are executed multiple times, speculating on the possible orderings eventually determined by the OAB service. We formalize the Speculative Transactional Replication (STR) problem by means of a set of properties ensuring that transactions are never activated on inconsistent snapshots, as well as the minimality and completeness of the set of speculatively explored serialization orders. Finally, we present a protocol solving the STR problem, along with simulation results assessing its effectiveness.

1. Introduction

Active Replication (AR) is a fundamental approach for achieving fault tolerance and high availability [1]. When applied to transactional systems, it requires that replicas agree on a common order for the execution of transactions. This is typically achieved by relying on some form of non-blocking distributed consensus, such as Atomic Broadcast [2] (AB), before transaction processing takes place [3,4].

Given that AB may exhibit non-negligible latency, it is important to design strategies to mitigate its impact. One of such strategies is based on the overlap of local processing and replica coordination [5]. This can be achieved using an Optimistic Atomic Broadcast (OAB) service, that provides an "early" (though potentially erroneous) guessing of the final outcome of the coordination phase [6]. Exploiting the optimistic message delivery order, each site may immediately start the (optimistic) processing of transactional requests without waiting for the completion of the coordination phase.

Clearly, this strategy pays off only if the final total order does not contradict the initial optimistic guess. Further, existing OAB-based solutions only permit the parallel activation of optimistically delivered transactions that are known not to conflict with each other [5,7]. Such a choice simplifies the management of local processing activities, avoiding the risks of propagating the results generated by optimistically delivered transactions. However, it can seriously constrain the achievable degree of parallelism [8]. Additionally, existing approaches require *a priori* knowledge of both read and write sets associated with incoming transactions in order to frame transactions within conflict classes as soon as they are delivered optimistically, i.e., prior to their execution. This requirement raises the non-trivial problem of systematically predicting data access pat-

* Corresponding author.
E-mail addresses: romanop@gsd.inesc-id.pt (P. Romano), palmieri@dis.uniroma1.it (R. Palmieri), quaglia@dis.uniroma1.it (F. Quaglia), nonius@gsd.inesc-id.pt (N. Carvalho), ler@ist.utl.pt (L. Rodrigues).

0022-0000/$ – see front matter
http://dx.doi.org/10.1016/j.jcss.2013.07.006

terns, which may entail significant overestimation of the likelihood of transaction conflicts, with an obvious negative impact on concurrency, especially in case of transactions exhibiting non-deterministic data access patterns.

The above drawbacks are exacerbated in a number of realistic scenarios such as large scale geographical replication, where guessing the final order can be very challenging [9], or in systems where the ratio between the communication delay and the computation granularity is very large, such as Transactional Memories (as discussed, for instance, in [10]).

In this paper, we address these challenges by exploring the use of speculative transaction processing. Particularly, we investigate, from a theoretical perspective, the issues related to the adoption of a speculative approach to replication of transactional systems, which we call Speculative Transactional Replication (STR). The idea underlying STR is rather simple: exploring multiple serialization orders for the optimistically delivered transactions, letting them observe the data item versions generated by conflicting transactions, rather than pessimistically blocking them waiting for the outcome of the coordination phase.

We frame the problem in a (desirable) model in which we do not assume the availability of any *a priori* information on the set of data items to be accessed by the transactions (in either read or write mode), and in which we allow data access patterns to be influenced by the state observed during the execution. Next, we formalize a set of correctness criteria for the speculative exploration of the permutations of the optimistically delivered transactions, demanding the *on-line* identification of all and only the transaction serialization orders that would cause the optimistically executed transactions to exhibit distinct outcomes.

Also, we present an STR protocol relying on a novel graph-based construct, named Speculative Polygraph (SP), which encodes information on the conflict relations materialized during the speculative execution of transactions. SPs are designed to identify what subsets of the speculatively available data item versions would be visible in any view-serializable execution, thus ensuring *completeness* and *non-redundancy* of the set of explored speculative serialization orders.

To assess the viability and practical relevance of our proposal, we also report simulation results based on data access patterns produced by a well-known benchmark for Software Transactional Memories [11].

The remainder of this paper is structured as follows. In Section 2 we discuss related work. Section 3 illustrates the system model. In Section 4 we formalize the properties characterizing STR. In Section 5 we introduce the Speculative Polygraph construct. Section 6 presents the STR protocol, along with the proof of its correctness. Finally, Section 7 provides the results of the simulation study.

2. Related work

Atomic Broadcast (AB) based replication protocols for transactional systems [3–5,12,13] achieve global transaction serialization order (across all the replicas) without incurring the scalability problems affecting classical eager replication mechanisms based on distributed locking and atomic commit protocols [14]. Our STR protocol builds on Optimistic Atomic Broadcast (OAB) [5,6] and, compared to the above results, takes a more aggressive optimistic approach by speculatively exploring the minimum set of serialization orders in which optimistically delivered transactions observe every distinct snapshot of the transactional system's state. Further, unlike other OAB-based schemes [5], STR does not rely on the assumption of *a priori* knowledge of the data items to be accessed by transactions, and makes use of optimistic transaction scheduling to favor concurrency.

To the best of our knowledge the idea of exploiting speculation in transaction processing environments has been first investigated in [15] and [16]. The work by Bestavros and Braoudakis [15] targets *non-replicated* real-time databases and shows the benefits, in terms of transaction timeliness, by speculatively forking, upon detection of a conflict, a copy of the current transaction that remains idle and serves as a save-point to reduce the rollback cost. On the other hand, STR addresses speculation in the context of OAB-based replication of transactional systems, and permits to concurrently process multiple instances of a same transaction in different serialization orders. The solution by Reddy and Kitsuregawa [16] targets distributed databases relying on distributed locking and atomic commit for transaction validation. Further, it constrains the transaction execution model, making the assumption that each data item can be written by a transaction at most once.

3. System architecture

3.1. Distributed system model

We consider a classical distributed system model [17] consisting of a set of processes $\Pi = \{p_1, \ldots, p_n\}$ communicating via message passing, which can fail according to the fail-stop (crash) model.

We assume that the number of correct processes (i.e., processes that do not fail) and the system synchrony level suffice to support an OAB service[1] providing the following interface: `TO-broadcast`(m), which broadcasts messages to all the processes in Π; `Opt-deliver`(m), which delivers message m to a process in Π in a tentative, also called optimistic, order; `TO-deliver`(m), which delivers a message m to a process in Π in a so-called *final order* that is the same for all the processes in Π. The OAB service is characterized by the following properties [6]:

[1] To this end, for instance, it is enough to assume an asynchronous distributed system model, augmented with an eventually perfect failure detector, and the correctness of a majority of processes [17].