

Exploiting Precision vs. Efficiency Tradeoffs in Symmetric Replication Environments

(Brief Announcement)

Uğur Çetintemel

Department of Computer Science
Brown University
ugur@cs.brown.edu

Peter J. Keleher

Department of Computer Science
University of Maryland
keleher@cs.umd.edu

Yanif Ahmad

Department of Computer Science
Brown University
yna@cs.brown.edu

1. Introduction

In this paper, we address (1) numerical data that are replicated and updated at multiple network locations (which we refer to as *symmetric* replication); (2) environments where maintaining strict data consistency is prohibitive due to large system scale, high volume of updates, or communication restrictions; and (3) applications that can tolerate bounded imprecision in the data they observe. Wide-area network management, on-line commodity distribution, load balancing, and resource monitoring, etc., demonstrate these characteristics.

This paper describes the key features of *ReBound*, a system that uses efficient distributed precision control to support and exploit data precision vs. efficiency tradeoffs in symmetric replication environments. *ReBound* enables clients to specify read requests tagged with custom precision bounds, which are satisfied cooperatively by the servers. *ReBound* supports two types of reads. *Continuous* reads require that the data cached by the clients always meet the specified precision constraints. This is accomplished by a push-based approach where the servers refresh client caches with new updates as necessary. *Ad-hoc* reads, on the other hand, have one-time semantics and are realized using a pull-based approach where the clients pull the new updates from the servers.

Most previous work studied precision bounding in asymmetric replication (e.g., master-copy) environments. Early work that addressed symmetric replication environments commonly addressed general distributed constraint maintenance and employed expensive mechanisms that are impractical for the types of applications and environments we target. Recently Yu and Vahdat [2] described a practical precision control algorithm for replicated network services. Yu's algorithm basically works by *partitioning* the client-specified precision bounds across the servers in the system. Each server then efficiently bounds imprecision by limiting the updates that it commits and that are yet *unknown* (i.e., not yet propagated) to clients. Upon receiving an update, the server checks local criteria to decide whether the commitment of the update violates its partition of the precision bound (i.e., local bound). If the criteria are met, the server commits the update locally. Otherwise, the server performs synchronization and pushes the unknown updates to the proper clients.

ReBound generalizes and extends previous work with a new algorithm for continuous reads, support for ad-hoc reads, and lightweight adaptation mechanisms for coping with dynamically changing update load. In the next section, we discuss these features in more detail.

2. ReBound System Model

We now briefly outline *ReBound*'s key features:

Support for continuous reads. We propose two algorithms that maintain continuous precision bounds by properly controlling update commitment at servers. The first algorithm partitions the precision bounds across servers (based on per-server weights). Each server then maintains a local precision bound and ensures that the updates it commits do not invalidate its own local bound. This algorithm is a simple generalization of that presented in [2], which partitions bounds uniformly across servers. The second algorithm replicates the precision bounds across servers, establishing a global precision bound that is shared and maintained by all servers. The advantage of a shared global bound is the flexibility of using any "unused imprecision space" available in the system. On the other hand, this requires more server-server synchronization than the partitioning approach does, thereby reducing server autonomy.

Support for ad-hoc reads. *ReBound* also addresses and supports ad-hoc reads with precision bounds. Our algorithms exploit the already registered continuous precision bounds, if available, to efficiently select a proper subset of servers whose unknown updates need to be pulled by the client to satisfy the specified bound.

Lightweight adaptation mechanisms. *ReBound* employs a simple but effective adaptation mechanism. We use of per-replica weights, which define the autonomy of the servers in terms of the volume of updates they can commit locally. We then enable dynamic, pairwise redistribution of these weights to cope with changing update patterns across servers. This mechanism is similar to the sum-preserving weight distribution proposed in the context of Deno [1].

3. Status and future work

Our preliminary experimental results, based on a prototype implementation, verify the performance advantages of exploiting precision vs. efficiency tradeoffs. We are currently investigating proxy-based hierarchical organizations to effectively scale up to a large number of clients and data items. In this model, proxies are responsible for executing the requests of their client set by interacting with the servers (or higher-level proxies). We are developing adaptive policies for setting appropriate *aggregate* proxy precision bounds that balance server-proxy push and proxy-server pull, thereby optimizing overall communication requirements.

References

- [1] U. Çetintemel, P. J. Keleher, and M. J. Franklin. Support for Speculative Update Propagation and Mobility in Deno. In *IEEE Intl. Conf. on Distributed Computing Systems (ICDCS)*, Phoenix, 2001.
- [2] H. Yu and A. Vahdat. Efficient Numerical Error Bounding for Replicated Network Services. In *Proc. of the 26th VLDB Conf.*, Cairo, Egypt, 2000.