

Evaluating Total Order Algorithms in WAN

Tal Anker

School of Engineering and Computer Science
The Hebrew University of Jerusalem, Israel
anker@cs.huji.ac.il
Radlan Computer Communications, Israel
tala@radlan.com

Danny Dolev

Gregory Greenman
and Ilya Shnayderman
School of Engineering and Computer Science
The Hebrew University of Jerusalem, Israel
{dolev, gregory, ilia}@cs.huji.ac.il

Abstract—Agreed message delivery is an important service in distributed systems, especially when dealing with fault-tolerant applications. This work studies factors influencing protocols that provide global order in Wide Area Networks. Performance evaluation in real network conditions was conducted in order to compare two recently published algorithms. It was found that loss rate and variations in message propagation time have a significant impact on the performance of the algorithms.¹

Keywords: total order, atomic broadcast, group communication, WAN

I. INTRODUCTION

Message delivery order is a fundamental building block in distributed systems. For more than two decades, researchers have been developing efficient algorithms that offer useful guarantees. The agreed order is one of the basic message delivery order guarantees, allowing distributed applications to use the state-machine replication model to achieve fault tolerance and data replication. Extensive analysis of algorithms providing agreed message order can be found in [1]. In addition, two recently introduced algorithms have presented methods to implement an agreed message delivery order in a Wide Area Network (WAN): *Optimistic Algorithm* [2] and *Hybrid Algorithm* [3]. These algorithms give an early notification of message delivery order to the application, even before the final order of the messages is established. In [4], it is shown that this early notification is useful in replicated databases.

Replicated databases are often built atop Group Communication Systems (GCS). Such middleware systems usually provide an application builder with tools, such as message ordering, reliable delivery and group membership services. We have developed such a GCS for a WAN called Xpand [5]. In order to provide Xpand users with an efficient WAN-oriented agreed message order, we have evaluated recent WAN global ordering algorithms, namely, *Optimistic Algorithm* and *Hybrid Algorithm*. The assumptions used by these algorithms are suitable for Xpand. We evaluated the algorithms in order to implement them in Xpand and analyzed the factors which have substantial impact on algorithms' latency. Theoretical evaluation of distributed algorithms is complicated, due to multiple factors that have impact on algorithms' performance,

e.g. network topology, message frequency, failure model etc. Metrics commonly used for such comparisons (e.g. the number of rounds of message exchanges or the number of sent messages) proved not to be accurate enough in a WAN [6].

We studied the algorithms' latency in a steady state case in a real network and discovered that two factors, namely, a loss rate and variations in message propagation time, have a significant impact on algorithms' performance.

II. ALGORITHMS UNDER COMPARISON

When comparing ordering algorithms, it is important to consider the guarantees each algorithm provides. We start by presenting the order required by a replicated database application, namely, *Uniform Total Order (UTO)*. A formal definition of a UTO broadcast guarantee can be found in [3]. In essence, it provides the same message delivery order for all the processes, including the faulty ones².

Before a UTO is agreed on, a Preliminary Order (PO) is "proposed" by each of the processes. If the PO is identical for all correct (non-faulty) processes, it is called Total Order (TO). PO and TO should either be confirmed or changed by the UTO later. The algorithm presented in [3] (*Hybrid Algorithm*) provides both TO and UTO, while the algorithm presented in [2] (*Optimistic Algorithm*) provides PO (which the authors call Optimistic Order) and TO. Although no protocol is proposed for UTO, it could easily be built atop TO by collecting acknowledgments from the majority of the processes that have delivered messages in TO. The most important difference between Optimistic Order and TO is that in the latter the order may be changed *iff* a process is suspected to have failed, while Optimistic Order may be changed even if no process has been suspected. Below we discuss the protocols used by these algorithms in order to achieve the above ordering services.

In Optimistic Algorithm, a process is selected to serve as the Sequencer. For every message it receives, the Sequencer gives a TO number and notifies all the processes about the message order. The Optimistic Order is achieved by predicting (with a high probability) the time when the message is received by the Sequencer.

¹Slightly improved version of the paper which appeared at Workshop on Large-Scale Group Communication at SRDS 2003.

²We do not consider Byzantine Failures in this work.

In Hybrid Algorithm, each process is marked as either passive or active, according to the location of the process in the network and its message sending rate. Active processes act as sequencers for passive ones. Lamport symmetric protocol [7] is used to achieve TO in the set of active processes. In order to compare a sequencer-based approach with a symmetric approach, we chose the topology and process sending rates so that all the participating processes were marked as active ones. The latency of Lamport symmetric protocol [7] can be improved by using local clocks as was shown in [3].

III. METHODOLOGY

This section describes the methodology used for the evaluation of the ordering algorithms.

It is important to note that throughout the experiments we did not consider processes suspicions/failures. Although these are important factors, they should be studied in a separate work, since the current paper focuses on the steady state case. However, message losses do occur in the steady state and may have significant impact on the algorithms' performance [6]. In order to achieve TO/UTO guarantee, lost messages must be re-acquired. Xpand [5] guarantees reliability by retransmitting lost messages, thus providing an appropriate framework for this study.

In the experiments, we used nodes from the RON project ([8]). The tests involved 6 sites, 5 residing in North America and one in Jerusalem, Israel. While the number of sites is relatively small, Xpand's hierarchical structure and its support for multiple groups allow to extend the algorithms to large-scale systems. The nodes themselves were Celeron/733 or PentiumIII/1.1G machines running FreeBSD/Linux operating system, connected via commodity Internet and Internet2. The links had diverse delays and varying packet loss rates.

We used Xpand to collect information about the network characteristics. Each host generated a new data packet every 10ms. Although data packets were sent by UDP protocol, the rate was limited by Xpand's TCP-friendly congestion control [9] mechanism. Each node listed the time when a packet was received from the network. The ordering algorithms were evaluated by emulating their protocols behavior on the message logs.

A. Application Layer Multicast

Xpand reliable multicast service relies on the availability of a many-to-many communication substrate. The network service that supports many-to-many communication in IP networks is IP multicast. There is a limitation on using IP multicast, since it is only partially deployed in various networks, which creates isolated regions supporting IP multicast.

In order to overcome this difficulty and to "bridge" between regions supporting IP multicast, Xpand introduces a layer called *Application Layer Multicast* (ALM) which is used only as an interim solution. This layer constructs a message distribution tree incorporating representative nodes, each node being chosen from a region that supports a native multicast. The tree construction is performed automatically and is beyond the scope of this paper (See [5], [10]).

B. Clock Skews

In order to estimate the ordering algorithm latency, we needed a good evaluation of the clock differences among all the participating processes. This was achieved using the mechanism described in [6]. The assumption made in calculating the clock differences was that message Round Trip Time (RTT) in a WAN was stable and rarely changed. The RTT was also assumed to be symmetric. Although these assumptions may not hold for a general case, they were applicable in our experiments. Time drift may be another concern when evaluating time differences. In our experiments, the logs were collected over a relatively short interval (several minutes' span), thus making time drift between the clocks negligible.

C. Implementation and Optimization of the Algorithms

This section describes essential implementation and optimization details. In order to emulate the ordering protocols, we assumed that control information can be piggybacked on data messages. This technique did not introduce any additional delay into the ordering protocols latency, as data messages were generated at a high rate.

Optimistic Algorithm relies on delay calculations. In order to emulate this algorithm, we used the logs to estimate the delays, and then emulated the protocol running on the logs using the estimated delay. High message sending rate and small variation in message propagation time may cause Optimistic Order to wrongly predict TO (on messages received in the same time window). To improve the TO prediction, Optimistic Algorithm batches messages [2]. For the same purpose we used message-sending time.

Some applications may require messages to be delivered in the same order they have been sent (*FIFO*). Since this is not part of the UTO definition, our implementation of Optimistic Algorithm does not provide FIFO, in order to achieve an improved latency.

Since we achieved clock synchronization, in Hybrid Algorithm (See III-B) we could put a timestamp on messages and use it to achieve TO.

In order to reduce the impact of message losses, we duplicated previous acknowledgments on each data message. This is feasible since (1) the size of an acknowledgment is very small and (2) an acknowledgment on message m also acknowledges messages preceding m .

IV. PERFORMANCE RESULTS

The logs were collected in multi-continent (WAN) setting of RON Testbed Network. The experiment included six sites: Zikit1 (Jerusalem), RON0 (Boston), RON2 (Salt Lake City), RON33 (Ottawa), RON35 (Berkeley), RON45 (Chicago).

As there is no IP multicast connectivity among these sites, a proprietary application layer multicast (ALM) mechanism was used [5]. The obtained message distribution tree is presented in Figure 1. RON33 was selected to be the tree root by the algorithm used in the dynamic ALM tree construction and was used as the Sequencer for Optimistic Algorithm.

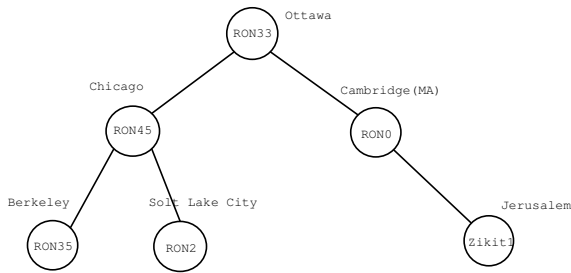


Fig. 1. ALM Layout

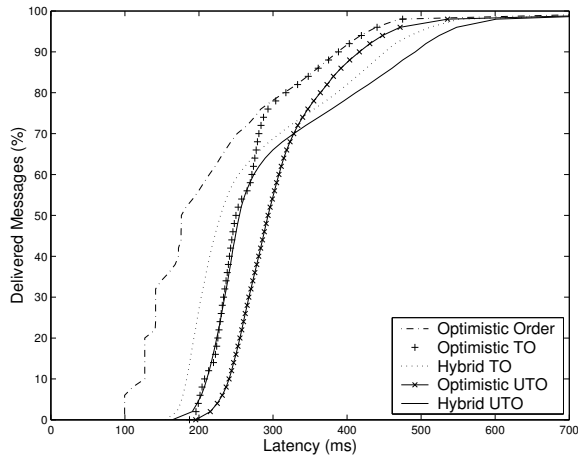


Fig. 2. Comparing the Algorithms

The maximal RTT registered between RON2 and Zikit1 was 330ms. It is worth noting that in some links a significant percentage of messages experienced substantial delays. Also, we noticed that loss rate varied significantly over different links (the worst link experienced the average of 3.6% loss rate).

In order to compare the algorithms, we studied the latencies. For Optimistic Algorithm, we measured the time between message sending and its delivery by all the processes (*message latency*) in Optimistic Order, TO and UTO. For Hybrid Algorithm, we measured the latencies of TO and UTO. For some messages, very high latencies were observed due to loss bursts. Therefore, the average values are not representative, and we chose to use Cumulative Distribution Function (CDF) (Figure 2). Here, the X axis presents latencies, while the Y axis shows the percentage of messages delivered by specific time.

As can be seen from the data represented in the graph, approximately at the 60% level, the behavior of the curves changes. Below this point, the latencies are similar to those expected. A message m is deliverable in Optimistic TO only after m 's TO number has been received from the Sequencer. Hybrid TO delivers m when a subsequent message from each process is received. Therefore, Hybrid TO imposed a shorter delay on m , as the sending rate was high during the

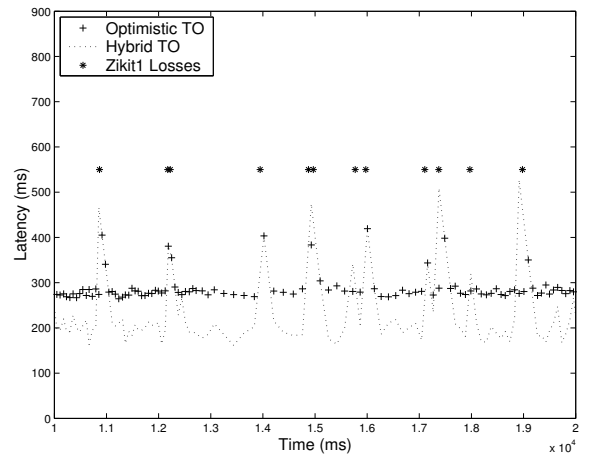


Fig. 3. Impact of Losses

experiment. Optimistic Order latency is better than that of Hybrid TO, since it waits only for a precomputed delay to deliver m and does not require reception of any additional message.

In the upper part of the graph, the pattern of the curves changes drastically. Optimistic Order curve converges with that of Optimistic TO. It means that Optimistic Order was predicted wrongly, hence, m 's Optimistic Order delivery time was substituted by Optimistic TO delivery time. The decrease in Optimistic TO slope indicates an increase in the number of messages delivered with a high latency. This increase is even more significant in Hybrid TO.

As we found out, this slowdown of TO algorithms was caused by message losses. The graph in Figure 3 shows the latencies measured by Zikit1 for its own messages. Only a short time interval is presented (time is measured from the start of the experiment) in order to make the graph more comprehensible. It is evident that message losses have significant influence on the latency. It is worth noting that the impact on Hybrid TO is higher than on Optimistic TO. In order to deliver message m , the Lamport Algorithm (used in Hybrid TO) requires that a message with a timestamp higher than m be received from every process, while keeping the FIFO order. To understand why losses increase the Optimistic TO latency, we need to consider message completion mechanism in Xpand [5]. This mechanism causes a lost message m to be retransmitted point-to-point, and consequently, the Sequencer receives m much quicker than Zikit1 and then sends its TO number to all the nodes. This causes Zikit1 to postpone the delivery of the messages following m , until m is recovered.

In Figure 4 the reasons for Optimistic Order delivery slowdown are shown. A replicated database may only benefit from correctly "guessed" Optimistic Order. In case of a wrong guess, hereafter referred to as *miss*, Optimistic TO delivery time is to be used. If TO is not equal to Optimistic Order, the replicated database has to perform a rollback. We assumed that this rollback may be performed immediately after TO is received. Optimistic Order latency measured by Zikit1 for its

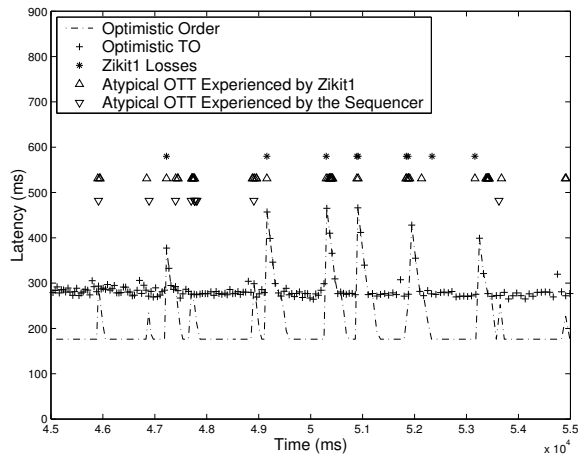


Fig. 4. Impact of OTT Variations

own messages is shown in Figure 4.

The percentage of misses varied from 11% measured by RON0 to 43% measured by RON35 for Optimistic Order³. It is noteworthy that once an application learns that Optimistic Order is incorrect, it has to perform the rollback for all the messages received starting from the missing message⁴. The peaks on the graph correspond to the first miss. Afterwards, Optimistic Order latency decreases for each following message, due to the reduction of the time interval from the moment when a message was sent till it is TO-delivered.

Although the numbers of misses might seem very high, it is not of great significant, as they largely depend on the rollback model and message sending rate. However, it is important to understand the reasons which caused those misses. As it can be seen in Figure 4, those misses are not evenly distributed over time, but occur in ranges. Such range is initiated by either a message loss or high (*atypical*) One-way Trip Time (*OTT*) experienced by Zikit1 as well as the Sequencer. Such OTT's, in turn, may be caused both by the message completion protocol and by the network. To study the network links' characteristics, we used *ping* utility and found out that RTT's of some messages were significantly higher than the average RTT on the same link. On some links, the number of atypical RTT's was as high as 2%.

V. CONCLUSIONS AND FUTURE WORK

The paper focuses on factors that have considerable impact on the Total Order algorithms' latency in a WAN. The algorithms under study were found to be vulnerable to message losses and to variations in message propagation time which are inherent in a WAN environment. Hence, while developing a distributed algorithm for a WAN, it is essential to consider the impact of those factors on the performance. We also found out that the message completion mechanism in Xpand instigates misses in Optimistic Order and should be modified.

³The misses were not registered by Hybrid Algorithm as no failures happened.

⁴Our model considers all messages sent to a single group as potentially conflicting transactions [4].

Another important observation is that the loss rate of a link does not vary considerably. This fact was also mentioned in [11]. We believe that an efficient protocol in WAN is to combine the usage of unreliable channels built over lossy links with reliable channels over lossless links.

In order to further improve the performance of the studied algorithms, some optimizations are to be carried out, that require from a process to measure loss rate on its links. In case when a receiver finds out that the loss rate of incoming messages is too high, it is to stop delivering messages in Optimistic Order. In the other case, when a sender finds out that its messages are frequently lost by other processes Hybrid Algorithm is to mark the sender as passive. (See II.) Optimistic Algorithm is to use a similar approach. As an alternative Forward Error Correction code can be used to minimize the impact of message losses.

The study strategy is comprehensive enough to be extended to cover various network topologies, sending rates and other total order algorithms.

The set of collected logs reflecting the real network conditions, as well as the simulator designed for this study, allow an algorithm developer to evaluate the performance. The logs and the simulation code are located at www.cs.huji.ac.il/labs/danss/TO_logs.html

REFERENCES

- [1] X. Defago, A. Schiper, and P. Urban, "Totally ordered broadcast and multicast algorithms: a comprehensive survey," 2000.
- [2] F. Moura A. Sousa, J. Pereira and R. Oliveira, "Optimistic total order in wide area networks," in *Proc. 21st IEEE Symposium on Reliable Distributed Systems*. October 2002, pp. 190–199, IEEE CS.
- [3] P. Vicente and L. Rodrigues, "An indulgent uniform total order algorithm with optimistic delivery," in *Proc. 21st IEEE Symposium on Reliable Distributed Systems*. October 2002, IEEE CS.
- [4] B. Kemme, F. Pedone, G. Alonso, and A. Schiper, "Processing transactions over optimistic atomic broadcast protocols," in *Proceedings of 19th International Conference on Distributed Computing Systems (ICDCS'99)*, 1999.
- [5] T. Anker, G. Chockler, I. Shnayderman, and D. Dolev, "The Design and Performance of Xpand: A Group Communication System for Wide Area Networks," Tech. Rep. 2001-56, Institute of Computer Science, The Hebrew University of Jerusalem, Jerusalem, Israel, August 2001, URL: <http://leibniz.cs.huji.ac.il/research/>, See also the previous version TR2000-31.
- [6] O. Bakr and I. Keidar, "Evaluating the running time of a communication round over the internet," in *21th Annual ACM Symposium on Principles of Distributed Computing (PODC)*, 2002, pp. 243–252.
- [7] L. Lamport, "Time, Clocks, and the Ordering of Events in a Distributed System," *Comm. ACM*, vol. 21, no. 7, pp. 558–565, July 78.
- [8] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar, "An integrated experimental environment for distributed systems and networks," in *Proc. of the Fifth Symposium on Operating Systems Design and Implementation*, Boston, MA, Dec. 2002, USENIX Association, pp. 255–270, url: www.emulab.net.
- [9] I. Shnayderman T. Anker, D. Dolev and I. Sukhov, "Tcp-friendly many-to-many end-to-end congestion control," in *Proc. 22st IEEE Symposium on Reliable Distributed Systems*. October 2003, IEEE CS, To Appear.
- [10] T. Anker, D. Dolev, and I. Shnayderman, "Ad Hoc Membership for Scalable Applications," in *16th Intl. Conference on Distributed Computing Systems*, Oct. 2002.
- [11] Vern E. Paxson, *Measurements and Analysis of End-to-End Internet Dynamics*, Ph.D. thesis, University of California, Lawrence Berkeley National Laboratory, April 1997.