Available at
www.ComputerScienceWeb.com
POWERED BY SCIENCE @ DIRECT®

Theoretical Computer Science 299 (2003) 289–306

Theoretical Computer Science

www.elsevier.com/locate/tcs

Diffusion without false rumors: on propagating updates in a Byzantine environment

Dahlia Malkhi[a], Yishay Mansour[b], Michael K. Reiter[c]

[a] *School of Computer Science and Engineering, The Hebrew University of Jerusalem, Ross Building Givat Ram Campus, 91904 Jerusalem, Israel*
[b] *School of Mathematical Sciences, Tel Aviv University, Ramat Aviv, Tel Aviv 69978, Israel*
[c] *Department of Electrical and Computer Engineering and Department of Computer Science, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213–3891, USA*

Received 1 August 2000; received in revised form 27 April 2001; accepted 26 March 2002
Communicated by D. Peleg

Abstract

We study how to efficiently diffuse updates to a large distributed system of data replicas, some of which may exhibit arbitrary (Byzantine) failures. We assume that strictly fewer than t replicas fail, and that each update is initially received by at least t correct replicas. The goal is to diffuse each update to all correct replicas while ensuring that correct replicas accept no updates generated spuriously by faulty replicas. To achieve this, each correct replica further propagates an update only after receiving it from at least t others. In this way, no correct replica will ever propagate or accept an update that only faulty replicas introduce, since it will receive that update from only the $t-1$ faulty replicas.

We provide the first analysis of diffusion protocols for such environments. This analysis is fundamentally different from known analyses for the benign case due to our treatment of fully Byzantine failures—which, among other things, precludes the use of digital signatures for authenticating forwarded updates. We propose two measures that characterize the efficiency of diffusion algorithms, *delay* and *fan-in*, and prove general lower bounds with regards to these measures. We then provide a family of diffusion algorithms that have nearly optimal delay/fan-in product.

1. Introduction

A diffusion protocol is the means by which an update initially known to a portion of a distributed system is propagated to the rest of the system. Diffusion is useful for

E-mail addresses: dalia@cs.huji.ac.il (D. Malkhi), mansour@math.tau.ac.il (Y. Mansour), reiter@cmu.edu (M.K. Reiter).

0304-3975/03/$ - see front matter © 2002 Elsevier Science B.V. All rights reserved.
PII: S0304-3975(02)00325-0

driving replicated data towards a consistent state over time, and has found application for this purpose, e.g., in USENET News [13], and in the Grapevine [2] and Clearinghouse [26] systems. The quality of a diffusion protocol is typically defined by the delay until the update has reached all replicas, and the amount of message traffic that the protocol generates. It is desirable for a diffusion method to incur low communication costs, and for the delay to be small so as to limit the storage consumed by updates during their diffusion.

In a large and highly decentralized computer network, the diffusion of messages is like spreading news or rumors in the real world. In both realms, an initial group of participants receives a (news) update directly from a trustworthy source. Updates then spread via media and rumor, where spurious updates or alterations to updates might be introduced. In this paper, we provide the first study of update diffusion in these settings. More specifically, we study the diffusion of updates in distributed systems where components can suffer arbitrary (Byzantine) failures.

The framework for our study is a network of n data replicas, of which strictly less than some threshold t can fail arbitrarily, and to which updates are introduced continually over time. For example, these updates may be sensor readings of some data source that is sampled by replicas, or data that the source actively pushes to replicas. However, each update is initially received only by a random subset of the correct replicas of some size $\alpha \geqslant t$, and so replicas engage in a diffusion protocol to propagate updates to all correct replicas over time.

In our study, we allow fully Byzantine failures, and thus cannot rely on digital signatures to authenticate the original source of a message that one replica forwards to others. While maximizing the fault models to which our upper bounds apply, avoiding digital signatures also strengthens our results in other respects. First, in a network that intrinsically provides the correct sender address for each message due to the presumed difficulty of forging that address, avoiding digital signatures avoids the administrative overheads associated with distributing cryptographic keys. Second, even when the sender of a message is not reliably provided by the network, the sender can be authenticated using techniques that require no cryptographic assumptions (for a survey of these techniques, see [29]). Employing digital signatures, on the other hand, would require assumptions limiting the computational power of faulty replicas. Third, pairwise authentication typically incurs a low computation overhead on replicas, whereas digitally signing each message would impose a significantly higher overhead.

For our study of diffusion in a Byzantine setting, we assume round-based protocols, in which all replicas operate in synchronous rounds, in each of which a correct replica is allowed to send at most F^{out} message (the “fan-out”). We then focus on a particular type of protocols, in which any replica that does not obtain the update directly from the source must receive copies of the update directly from at least t different replicas before it “accepts” the update as one actually generated by the source (as opposed to one generated spuriously by a faulty replica). The justification for this approach is that no non-faulty replica will ever echo a bogus update, because the first such replica to do so must receive t distinct copies of the update, which cannot be generated by the $t-1$ Byzantine replicas. We start our analysis by proposing two measures of quality: The first one, *delay*, is the expected number of rounds until any individual update is

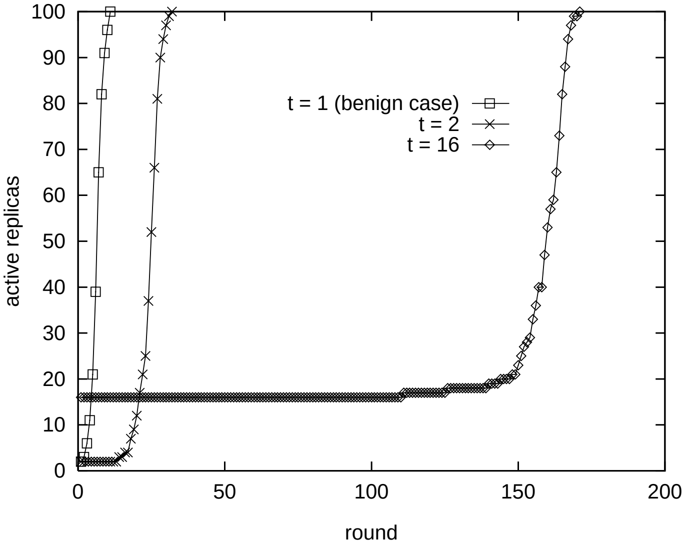

Fig. 1. Delay of random propagation with $n = 100$, $\alpha = t + 1$, $F^{\text{out}} = 1$.

accepted by all correct replicas in the system. The delay measure expresses the speed of propagation. The second, *fan-in*, is the expected maximum number of messages received by any replica in any round from correct replicas. Fan-in is a measure of the load inflicted on individual replicas in the common case, and hence, of any potential bottlenecks in execution. We then prove a lower bound of $\Omega((t \log(n/\alpha))/F^{\text{out}})$ on the delay of any diffusion protocol. We also show an inherent tradeoff between good (low) latency and good (low) fan-in, namely that their product is at least $\Omega(tn/\alpha)$.

To achieve efficient diffusion in our framework, we then suggest a family of round-based algorithms called ℓ-Tree. In these algorithms, replicas are arranged in a logical tree structure, with ℓ replicas in every node. Each replica sends messages at random to other replicas in its node, to its child nodes and to the root. We study the properties of ℓ-Tree and demonstrate that it covers much of the spectrum of optimal-delay protocols for their respective fan-in, to within logarithmic factors.

We emphasize that our treatment of full Byzantine failures renders our problem fundamentally different from the case of crash failures only. Intuitively, any diffusion process has two phases: in the first phase, the initially active replicas for an update send this update, while the other replicas remain inactive. This phase continues while inactive replicas have fewer than t messages. In the second phase, new replicas become active and propagate updates themselves, resulting in an exponential growth of the set of active replicas. To stress this fundamental difference, in Fig. 1 we depict the progress of "random" diffusion, in which each replica in each round chooses other replicas at random and sends them messages. The figure shows the number of active replicas plotted against round number, for a system of $n = 100$ replicas with different values of t, where $\alpha = t + 1$. The case $t = 1$ is indistinguishable from diffusion with benign failures only, since a single update received by a replica immediately turns it into an active one. Thus, in this case, the first phase is degenerate, and the exponential-growth phase occurs from the start. Previous work has analyzed the diffusion process in that case, proving propagation delay [7] that is logarithmic in the number of replicas. However, in the case that we consider here, i.e., $t \geqslant 2$, the delay is dominated by the initial phase. This dominant part of the delay will be manifested more precisely in the analysis we provide below.

The rest of the paper is organized as follows. In Section 1.1 we illustrate specific applications for which Byzantine message diffusion is suitable, and which motivated our study. We discuss related work in Section 1.2. In Section 2 we lay out assumptions and notation used throughout the paper, and in Section 3 we define our measures of diffusion performance. In Section 4 we provide general theorems regarding the delay and fan-in of diffusion protocols. In Section 5 we introduce the ℓ-Tree protocol family and analyze its properties. We summarize and discuss our results in Section 6. The expiration of updates from the propagation protocol is discussed in Section 7.

1.1. Motivation

The motivating application of our work on message diffusion is a data replication system called Fleet [18,19]. Fleet replicates data so that it will survive even the malicious corruption of some data replicas, and does so using adaptations of quorum systems to such environments [17]. A characteristic of these replication techniques that is important for this discussion is that each update is sent to only a relatively small subset (quorum) of servers, but one that is guaranteed to include t correct ones, where the number of faulty replicas is assumed to be less than t. Thus, after an update, most correct replicas have not actually received this update, and indeed any given correct replica can be arbitrarily out-of-date.

While this local inconsistency does not impact the global consistency properties of the data when the network is connected (due to the properties of the quorum systems we employ), it does make the system more sensitive to network partitions. That is, when the network partitions—and thus either global data consistency or progress of data operations must be sacrificed—the application may dictate that data operations continue locally even at the risk of using stale data. To limit how stale local data is when the network partitions, we use a diffusion protocol while the network is connected to propagate updates to all replicas, in the background and without imposing additional overhead on the critical path of data operations. In this way, the system can still efficiently guarantee strict consistency in case a full quorum is accessed, but can additionally provide relaxed consistency guarantees when only local information is used.

Another variation on quorum systems, *probabilistic quorum systems* [21], stands to benefit from properly designed message diffusion in different ways than above. Probabilistic quorum systems are a means for gaining dramatically in performance and resilience over traditional (strict) quorum systems by allowing a marginal, controllable probability of inconsistency for data reads. When coupled with an effective diffusion technique, the probability of inconsistency can be driven toward zero when updates are sufficiently dispersed in time.

More generally, diffusion is a fundamental mechanism for driving replicated data to a consistent state in a highly decentralized system. Our study sheds light on the use of diffusion protocols in systems where arbitrary failures are a concern, and may form a basis of solutions for disseminating critical information in survivable systems (e.g., routing table updates in a survivable network architecture).

1.2. Related work

The style of update diffusion studied here has previously been studied in systems that can suffer benign failures only. Notably, Pittel in [29] and Demers et al. [7] performed a detailed study of epidemic algorithms for the benign setting, in which each update is initially known at a single replica and must be diffused to all replicas with minimal traffic overhead. One of the algorithms they studied, called *anti-entropy* and apparently initially proposed in [2], was adopted in Xerox's Clearinghouse project (see [7]) and the Ensemble system [1]. Similar ideas also underly IP-Multicast [6] and MUSE (for USENET News propagation) [13]. This anti-entropy technique forms the first stage of the ℓ-Tree method that we study here. As described previously, however, the analysis provided here of the epidemic-style update diffusion is fundamentally different for Byzantine environments than for environments that suffer benign failures only, and provides the first study of diffusion in such environments.

Prior studies of update diffusion in distributed systems that can suffer Byzantine failures have focused on single-source broadcast protocols that provide reliable communication to replicas and replica agreement on the broadcast value (e.g., [12,8,3,16]), sometimes with additional ordering guarantees on the delivery of updates from different sources (e.g., [28,4,5,24,11]). The problem that we consider here is different from these works in the following ways. First, in these prior works, it is assumed that one replica begins with each update, and that this replica may be faulty—in which case the correct replicas can agree on an arbitrary update. In contrast, in our scenario we assume that at least a threshold $t>1$ of *correct* replicas begin with each update, and that only these updates (and no arbitrary ones) can be accepted by correct replicas. Second, these prior works focus on certain reliability, i.e., guaranteeing that all correct replicas (or all correct replicas in some agreed-upon subset of replicas) receive the update. Our protocols diffuse each update to all correct replicas only with some probability that is determined by the number of rounds for which the update is propagated before it is discarded. Our goal is to analyze the number of rounds until the update is expected to be diffused globally and the load imposed on each replica as measured by the number of messages it receives in each round.

This paper is based on our initial conference publication [14]. Since that time, several subsequent works have emerged. Optimized diffusion algorithms are provided in [19], that are specifically suitable for our motivating survivable replication application. A different diffusion paradigm is presented in [15] and independently in [23]. This paradigm forms propagation paths with messages before they are accepted as authentic. The algorithm in [15] propagates updates in time logarithmic in the number of replicas and linear in the number of corrupt replicas. A matching lower bound for this problem is also provided. The main price paid in the new protocol is in the size of messages used in the protocol and algorithmic complexity incurred in processing messages. Minsky [23] suggests methods for pruning the amount of information that is propagated, and provides simulation measurements that indicate that this complexity may be considerably decreased. Finally, a survivable replication system that makes use of diffusion for spreading updates asynchronously is suggested in [22].

2. System model

We assume a system of n replicas, denoted $p_1, \dots, p_n$. A replica that conforms to its I/O and timing specifications is said to be *correct*. A *faulty* replica is one that deviates from its specification. A faulty replica can exhibit arbitrary behavior (Byzantine failures). We assume that strictly fewer than t replicas fail, where t is a globally known system parameter.

Replicas can communicate via a completely connected point-to-point network. Communication channels between correct replicas are reliable and authenticated, in the sense that a correct replica p_i receives a message on the communication channel from another correct replica p_j if and only if p_j sent that message to p_i. Moreover, we assume that communication channels between correct replicas impose a bounded latency Δ on message transmission; i.e., communication channels are *synchronous*. Our protocols will also work to diffuse updates in an asynchronous system, but in this case we can provide no delay or fan-in analysis. Thus, we restrict our attention to synchronous systems here.

Our diffusion protocols proceed in synchronous rounds. A system parameter, *fan-out*, denoted F^{out}, bounds from above the number of messages any correct replica sends in a single round. A replica receives and processes all messages sent to it in a round, before the next round starts. Thus, rounds begin at least Δ time units apart.

Each update u is introduced into the system at a set I_u of randomly selected $\alpha \geqslant t$ *correct* replicas, and possibly also at some other, faulty replicas. We assume that all replicas in I_u initially receive u simultaneously (i.e., in the same round). The goal of a diffusion protocol is to cause u to be *accepted* at all correct replicas in the system. The class of protocol analyzed here works as follows: The update u is accepted at correct replica p_i if $p_i \in I_u$ or p_i has received u directly from t other distinct replicas. If p_i has accepted u, then we also say that p_i is *active* for u (and is *passive* otherwise). In all of our diffusion protocols, we assume that each message contains all the updates known to the sender, though in practice, obvious techniques can reduce the actual number of updates sent to necessary ones only.

3. Measures

We study two complexity measures: *delay* and *fan-in*. For each update, the delay is the worst-case expected number of rounds from the time the update is introduced to the system until all correct replicas accept the update. Formally, let η_u be the round number in which update u is introduced to the system, and let τ_p^u be the round in which a correct replica p accepts update u. The delay is $E[\max_{p,C}\{\tau_p^u\} - \eta_u]$, where the expectation is over the random choices of the algorithm and the maximization is over correct replicas p, all failure configurations C containing fewer than t failures, and all behaviors of those faulty replicas. In particular, $\max_{p,C}\{\tau_p^u\}$ is reached when the faulty replicas send no updates, and so this is the behavior we assume from them when analyzing delay.

We define Fan-in, denoted by F^{in}, to be the expected maximum number of messages that any correct replica receives in a single round from correct replicas under all possible failure scenarios. Formally, let ρ_p^i be the number of messages received in round i by replica p from correct replicas. Then the fan-in in round i is $E[\max_{p,C}\{\rho_p^i\}]$, where the maximum is taken with respect to all correct replicas p and all failure configurations C containing fewer than t failures. An amortized fan-in is the expected maximum number of messages received over multiple rounds, normalized by the number of rounds. Formally, a k-amortized fan-in starting at round l is $E[\max_{p,C}\{\sum_{i=l}^{l+k} \rho_p^i/k\}]$. We emphasize that fan-in and amortized fan-in are measures only for messages from correct replicas.

4. General results

In this section we present general results concerning the delay and fan-in of diffusion algorithms within our specification. Our first result is a lower bound on delay, that stems from the restriction on fan-out, F^{out}.

Theorem 4.1. *The delay of any diffusion algorithm adhering to our specification A is* $\Omega((t\log(n/\alpha))/F^{\text{out}})$.

Proof. Let u be any update, and let m_k denote the total number of times u is sent by correct processes in rounds $\eta_u+1,\dots,\eta_u+k$ in A. Denote by α_k the number of correct replicas that have accepted update u by the time round η_u+k completes. Since t copies of update u need to reach a replica (not in I_u) in order for it to accept the update, we have that $\alpha_k \leqslant \alpha + m_k/t$. Furthermore, since at most $F^{\text{out}}\alpha_k$ new updates are sent by correct processes in round η_u+k+1, we have that $m_{k+1} \leqslant m_k + F^{\text{out}}\alpha_k \leqslant F^{\text{out}}\sum_{j=0}^{k}\alpha_j$, where $\alpha_0=\alpha$. By induction on k, it can be shown that $\alpha_k \leqslant \alpha(1+F^{\text{out}}/t)^k$. Therefore, for $k<(t\log(n/\alpha))/F^{\text{out}}$ we have that $\alpha_k<n$, which implies that not all the replicas are active for update u. □

The next theorem shows that there is an inherent tradeoff between fan-in and delay.

Theorem 4.2. *Let A be any diffusion algorithm adhering to our specification. Denote by D its delay, and by* F^{in} *its D-amortized fan-in. Then* $DF^{\text{in}} = \Omega(tn/\alpha)$, *for* $t \geqslant 2\log n$.

Proof. Let u be any update. Since the D-amortized fan-in of A is F^{in}, with probability at least 0.9 (where 0.9 is arbitrarily chosen here as some constant between 0 and 1), the number of messages received (from correct replicas) by any replica in rounds $\eta_u+1,\dots,\eta_u+D < 10DF^{\text{in}}$. From now on we will assume that every replica p_j receives at most $10DF^{\text{in}}$ messages in rounds $\eta_u+1,\ldots,\eta_u+D$. This means that for each p_j, if p_j is updated by a set S_j of replicas during rounds $\eta_u+1,\ldots,\eta_u+D$, then $|S_j|\leqslant 10DF^{\text{in}}$. Some replica p_j must be the first (outside I_u) to become active for update u. This happens if out of the updates in S_j at least t updates are from replicas I_u, i.e., $|S_j\cap I_u|\geqslant t$. In order to show the lower bound, we need to exhibit an initial set I_u, such that if $10DF^{\text{in}}$ is too small then no replica becomes active. More specifically, for $D\leqslant \frac{1}{2}(nt/10F^{\text{in}}\alpha)$, we show that there exists a set I_u such that for each p_j, we have $|S_j\cap I_u|<t$.

We choose the initial set I_u as a random subset of $\{p_1,\ldots,p_n\}$ of size α. Let X_j denote the number of replicas in I_u from which messages are received by replica p_j during rounds $\eta_u+1,\ldots,\eta_u+D$, i.e., $X_j=|S_j\cap I_u|$. Since p_j receives at most $10DF^{\text{in}}$ messages in these rounds, we get

$$\begin{aligned}
Prob[X_j \geqslant k] &< \sum_{i=k}^{10DF^{\text{in}}} \frac{\binom{10DF^{\text{in}}}{i}\binom{n-10DF^{\text{in}}}{\alpha-i}}{\binom{n}{\alpha}}\\
&< \sum_{i=k}^{n} \binom{10DF^{\text{in}}}{i}\left(\frac{\alpha}{n}\right)^i\\
&\leqslant \left(\frac{10eDF^{\text{in}}\alpha}{kn}\right)^k c,
\end{aligned}$$

where the constant c is at most 2 if $D\leqslant\frac{1}{2}(nk/10eF^{\text{in}}\alpha)$, and hence we have that $Prob[X_j\geqslant t]<(1/2)^t$. By our assumption that $t\geqslant 2\log n$, we have that $Prob[X_j\geqslant t]<1/n^2$. This implies that the probability that all the X_j are at most t is at least $1-(1/n)$.

We have shown that for most subsets I_u if $D\leqslant\frac{1}{2}(nt/10eF^{\text{in}}\alpha)$ no new replica would become active. Therefore, for some specific I_u it also holds. (In fact it holds for most subsets.)

Recall that at the start of the proof we assumed that in D rounds no replica receives more than $10F^{\text{in}}D$ messages. This holds with probability at least 0.9. Therefore in most of the runs the delay is at least $\frac{1}{2}(nt/10eF^{\text{in}}\alpha)$, which implies that the expected delay is $\Omega(nt/F^{\text{in}}\alpha)$. □

5. ℓ-Tree propagation

In this section we introduce a family of algorithms, ℓ-Tree, which provide a spectrum of protocols that have optimal delay/fan-in tradeoff to within a logarithmic factor.

The ℓ-Tree method is quite simple to describe: We partition the replicas into blocks of size ℓ, where $\ell\geqslant 4t$, and arrange these blocks on the nodes of a binary tree. For each replica there are four interesting sets of replicas. The first set is the ℓ replicas at the root of the tree. The second set are the ℓ replicas in the node the replica is in (for simplicity it includes also the replica itself). The third and fourth sets are the ℓ replicas at the right and left children of the node that the replica is in. The total number of interesting replicas for each replica is at most 4ℓ, and we call it the *candidate set* of the replica. In each round, each replica chooses F^{out} replicas from its candidate set uniformly at random and sends a message to those replicas.

In the remainder of this section, we provide analysis of the fan-in and delay of the ℓ-Tree propagation method. In our analysis, we will highlight two extreme cases: On one end of the spectrum, we have n-Tree, i.e., $\ell = n$, in which case the method degenerates to an epidemic-style random propagation method. This method is similar to the "anti-entropy" method of [2,7]. The analysis below then provides, as a special case, measures for random propagation in a Byzantine setting. The results demonstrate that it has low fan-in and relatively large delay. At the other extreme, we have $4t$-Tree, i.e., $\ell = 4t$, which provides the opposite case, with high fan-in and low delay.

5.1. Fan-in

We begin with a study of the fan-in of the ℓ-Tree method.

Theorem 5.1. *The ℓ-Tree algorithm has fan-in $F^{\text{in}} = \mathrm{O}(nF^{\text{out}}/\ell)$ for $4t \leqslant \ell \leqslant nF^{\text{out}}/12 \log n$, and $F^{\text{in}} = \mathrm{O}(\log n)$ for $\ell > nF^{\text{out}}/12 \log n$.*

Proof. Clearly the highest fan-in is for replicas at the root node. We divide the analysis into two cases.

Case 1: Suppose $4t \leqslant \ell \leqslant nF^{\text{out}}/12 \log n$. Any replica at the root has a probability of at most $F^{\text{out}}/(2\ell)$ of receiving a message from any other replica. This implies that the expected number of messages per round is $nF^{\text{out}}/(2\ell)$. The probability that a replica receives more than $2F^{\text{out}}n/2\ell$ is at most $\mathrm{e}^{-F^{\text{out}}n/3(2\ell)}$ (using Chernoff bounds). Since $n \geqslant (12\ell/F^{\text{out}}) \log n$, the probability is at most $1/n^2$, and the bound follows.

Case 2: Now suppose $nF^{\text{out}}/12 \log n < \ell$. The probability that some replica at the root receives k or more messages in a round is bounded by

$$\binom{nF^{\text{out}}}{k}\left(\frac{1}{2\ell}\right)^k \leqslant \left(\frac{nF^{\text{out}}e}{2k\ell}\right)^k \leqslant \left(\frac{nF^{\text{out}}e}{2k(nF^{\text{out}}/12\log n)}\right)^k \leqslant \left(\frac{18\log n}{k}\right)^k.$$

For $k = 2 \times 18 \log n$, this probability is less than $1/n^2$, and hence the fan-in is bounded by $\mathrm{O}(\log n)$ in this case. □

The following corollary follows immediately from Theorem 5.1.

Corollary 5.2. *The ℓ-Tree algorithm has fan-in $\mathrm{O}(nF^{\text{out}}/\ell + \log n)$.*

Theorem 5.3. *The $\log n$-amortized fan-in of n-tree is $\mathrm{O}(F^{\text{out}})$.*

Proof. The probability that in $\log n$ rounds a specific replica receives more than $k = 6$ $F^{\text{out}} \log n$ messages is bounded by $\binom{nF^{\text{out}}\log n}{k}(1/n)^k$ which is bounded by $1/n^2$. The probability that some replica receives more than $k = 6F^{\text{out}} \log n$ messages is bounded by $1/n$. Thus, the $(\log n)$-amortized fan-in is at most $\mathrm{O}(F^{\text{out}})$. □

5.2. Delay

We now address the delay of ℓ-Tree. The analysis is done in two stages. The first step of the analysis is to calculate the expected delay until all the correct replicas in the root node become active. The second step is to add the delay of propagating updates down the tree.

Intuitive overview of delay analysis: We first shed intuition on the delay analysis with a "sloppy" proof sketch. Here, we assume $F^{\text{out}} = 1$ for simplicity. Additionally, we ignore message duplication for simplicity, and deal with the delay of each replica collecting *any* t messages. Both of these simplifying assumptions are relaxed in the precise analysis in Lemma 5.4 below, but do not substantially impact it.

The difficult part of the delay analysis is its first part. Consider the set of replicas consisting of those in the root node and those that are initially active. Let $\gamma \leqslant \ell + \alpha$ be the size of this set. Each replica targets the ℓ replicas in the root node with probability at least $1/4\ell$. The goal of the first part of the analysis is to determine the time it takes to activate the ℓ root node replicas. Following the first part, the analysis of propagation from the full-active root node down the tree easily yield an additive logarithmic element.

We now focus on the time is takes to activate the root node then. When α is a significant linear fraction of γ, say $\alpha \geqslant \gamma/4$, the derivation is easy: in each round, each replica receives at least an expected $\alpha(1/4\ell) \geqslant \frac{1}{16}$ messages. Hence, within an expected $O(t + \log \ell)$ rounds, all ℓ replicas in the root node become active.

However, when α is small relatively to γ, things are drastically different. Intuitively, this is because the α initial ones randomly target replicas, and it takes a long time for even single replica to be targeted t times, in order to collect t messages. In fact, a simplistic application of coupon collector's analysis here would yield that for any replica out of the ℓ, the expected time until it receives t different messages from the α active replicas is $(4\ell/\alpha)t$.

Fortunately, although the expected time for any particular replica to collect t messages is high, the expected time until *there exists* a (small) number of replicas that collect t messages is considerably smaller, when t is small. For example, consider the case $t = 2$. Then two initial replicas are sending messages at random, and after an expected $O(\sqrt{\ell})$ rounds, they collide with high probability (the "birthday" paradox). Hence, after $O(\sqrt{\ell})$ rounds, there exists a replica that receives $t = 2$ messages. More generally, our analysis shows that after $O(t(\ell/\alpha)^{(1-1/t)})$ rounds, there exist α root replicas that receive t messages each.

More specifically, we consider phases, where in each phase the number of active replicas is doubled. Let β denote the current number of active replicas, by m the number of messages sent by the active replicas, and by N_i^m the number of messages out of m received by replica i. Then the probability that replica i receives t messages is at least

$$Pr[N_i^m = t] \geqslant \binom{m}{t}\left(\frac{1}{4\ell}\right)^t\left(1 - \frac{1}{4\ell}\right)^{m-t} \geqslant \left(\frac{m}{t}\right)^t\left(\frac{1}{4\ell}\right)^t\left(\frac{1}{e}\right)^t,$$

where the last inequality uses the assumption that $m \leqslant 4t\ell$, for else, the trivial delay bound above follows. Hence, the expected number of replicas out of $\gamma - \beta$ that receive t messages is at least $(\gamma-\beta)(m/4et\ell)^t$. We are interested in the case $\beta \leqslant \gamma/2$ (when β reaches $\gamma/2$, we add a logarithmic additive, as indicated above). We have that for $m = (4\beta)^{1/t}16t\ell^{(1-1/t)}$, the expected number of replicas to receive t messages is then at least 2β. Hence, with high probability, the number of expected messages it takes β active replicas to double its size is at most m, and so the expected number of *rounds* is at most $m/\beta \leqslant 32t(\ell/\beta)^{(1-1/t)}$. It is rather simple to observe that the total number of rounds, when summed over all phases, will be dominated by the number of rounds in the first phase.

Detailed analysis: We now give a precise analysis of the delay, factoring in both F^{out} and the need to suppress duplicate messages (i.e., to collect t messages from *different* sources).

Let $R_{\beta,t}$ denote the expected number of steps needed to collect t *different* messages out of β ones, where at each step we sample a random message. This is the classical coupon collector problem, and we have (see [25, Chapter 3]):

$$R_{\beta,t} = \beta \sum_{j=\beta-t+1}^{\beta} 1/j \approx \beta \log \frac{\beta}{\beta - t + 1} + \mathrm{O}\left(\frac{\beta}{\beta - t+1}\right),$$

It is worth discussing how $R_{\beta,t}$ behaves for various values of β and t. For $\beta = t$ we have $R_{\beta,t} \approx t \log t$. For $\beta \geqslant 2t$ we have $R_{\beta,t} \leqslant 1.5t$. For all $\beta \geqslant t$, we have $R_{\beta,t} \geqslant t$. This implies that if the initial set size β is very close to t, then we have a slightly super-linear behavior of $R_{\beta,t}$ as a function of t, while if β is a fraction away from t then we have $R_{\beta,t}$ as a linear function in t.

Lemma 5.4. *The expected number of rounds of the ℓ-Tree algorithm until all the correct replicas in the root node are active is* $\mathrm{O}((R_{\alpha,t}/F^{\mathrm{out}})(\ell/\alpha)^{(1-1/2R_{\alpha,t})} + (t+\log(\ell))/F^{\mathrm{out}})$.

Proof. The outline of the proof is as follows. For the most part, we consider bounds on the number of messages sent, rather than directly on the number of rounds. It is more convenient to argue about the number of messages, since the distribution of the destination of each replica's next message is fixed, namely uniform over all candidate replicas. As long as we know that there are between α and 2α replicas active for u, we can translate an upper bound on the number of messages to an approximate upper bound on the number of rounds.

More specifically, let G denote the set of replicas that includes the correct replicas at the root node and the initially active replicas. Let γ denote G's size, i.e., $\ell \leqslant \gamma \leqslant \ell + \alpha$. Let $a(G)$ be the number of active replicas in G. By definition, initially we have $a(G) = \alpha$. So long as the number $\beta = a(G)$ satisfies $\alpha \leqslant \beta \leqslant \gamma/4$, we study $m^+(\beta)$, an upper bound on the number of messages needed to be sent such that with high probability, $1 - q^+(\beta)$, we have β new replicas in G change state to active. We then analyze the algorithm as composed of phases, where the jth phase starts with at least $\beta_j = 2^j\alpha$ active replicas. This implies that with probability $1-\delta$, $\delta = \sum_{j=0}^{k-1} q^+(2^j\alpha)$, after at most $\sum_{j=0}^{k-1} m^+(2^j\alpha)$ messages we have $a(G)\geqslant\gamma/2$, where $k=\log(\gamma/2\alpha)$. In such a case the number of rounds is bounded by $\sum_{j=0}^{k-1} m^+(2^j\alpha)/(2^jF^{\text{out}}\alpha)$.

At the end, we consider the case where $a(G)\geqslant\gamma/2$, and bound from above the number of rounds needed to complete the propagation algorithm. This case adds only an additive factor of $\mathrm{O}((t+\log\gamma)/F^{\text{out}})$ to the total delay.

We start with the analysis of the number of messages required to move from β active replicas in G to 2β, where $\beta\leqslant\gamma/4$. For any m, let N_i^m be the number of messages that p_i received, out of the first m messages. Let U_i^m be an indicator variable such that $U_i^m=1$ if p_i receives messages from t or more distinct replicas after m messages are sent, and $U_i^m=0$ otherwise.

We first use the *coupon collector*'*s* analysis to bound the probability that $U_i^m=1$ when N_i^m messages are received. Thus, a replica needs to get an expected $R_{\beta,t}$ messages before $U_i^m=1$, and so with probability $\leqslant 1/2$ it would need more than $2R_{\beta,t}$ messages to collect t different messages, i.e., $Prob[U_i^m=1|N_i^m=2R_{\beta,t}]\geqslant 1/2$.

Now, if m total messages are sent, we bound the probability that replica i receives $2R_{\beta,t}$ of them:

$$\begin{aligned}
Prob[N_i^m = 2R_{\beta,t}] &\geqslant \binom{m}{2R_{\beta,t}}\left(\frac{1}{4\ell}\right)^{2R_{\beta,t}}\left(1-\frac{1}{4\ell}\right)^{m-2R_{\beta,t}}\\
&\geqslant \left(\frac{m}{2R_{\beta,t}}\right)^{2R_{\beta,t}}\left(\frac{1}{4\ell}\right)^{2R_{\beta,t}}\left(\left(1-\frac{1}{4\ell}\right)^{m/2R_{\beta,t}}\right)^{2R_{\beta,t}}\\
&= \left(\frac{m\left(1-\frac{1}{4\ell}\right)^{m/2R_{\beta,t}}}{8\ell R_{\beta,t}}\right)^{2R_{\beta,t}}\\
&\geqslant \left(\frac{m(1-m/8\ell R_{\beta,t})}{8\ell R_{\beta,t}}\right)^{2R_{\beta,t}}\\
&\geqslant \left(\frac{m}{16\ell R_{\beta,t}}\right)^{2R_{\beta,t}},
\end{aligned}$$

where the last inequality holds for $m\leqslant 4\ell R_{\beta,t}$ (which trivially bounds m, as explained above). Putting the above together, we get that

$$\begin{aligned}
Prob[U_i^m = 1] &\geqslant Prob[N_i^m = 2R_{\beta,t}]Prob[U_i^m = 1|N_i^m = 2R_{\beta,t}]\\
&\geqslant \left(\frac{m}{16\ell R_{\beta,t}}\right)^{2R_{\beta,t}}\left(\frac{1}{2}\right).
\end{aligned}$$

Let U^m denote the number of nonactive replicas in G that received messages from t or more replicas after m messages are sent, i.e., $U^m=\sum_{i=\beta+1}^{\gamma} U_i^m$, where the active replicas are $p_1,\dots,p_\beta$. For $\beta \leqslant \gamma/4$ we have

$$\begin{aligned} E[U^m] &\geqslant (\gamma-\beta)\left(\frac{m}{16\ell R_{\beta,t}}\right)^{2R_{\beta,t}}\left(\frac{1}{2}\right)\\ &\geqslant \frac{\gamma}{4}\left(\frac{m}{16\ell R_{\beta,t}}\right)^{2R_{\beta,t}}\\ &\geqslant \frac{\ell}{4}\left(\frac{m}{16\ell R_{\beta,t}}\right)^{2R_{\beta,t}}. \end{aligned}$$

Our aim is to analyze the distribution of U^m. More specifically, we would like to find a bound $m^+(\beta)$ such that,

$$Prob[U^m \geqslant 2\beta] > 1 - q^+(\beta)$$

for any $m > m^+(\beta)$.

Generally, the analysis is simpler when the random variables are independent. Unfortunately, the random variables U_i^m are not independent, but using a classical result by Hoeffding [9, Theorem 4], the dependency works only in our favor. Namely, let X_i^m be i.i.d. binary random variables with $Prob[X_i^m = 1] = Prob[U_i^m = 1]$, and $X^m = \sum_{i=1}^n X_i^m$. Then,

$$Prob[U^m - E[U^m] \geqslant \delta] \leqslant Prob[X^m - E[X^m] \geqslant \delta].$$

From now on we will prove the bounds for X^m and they will apply also to U^m. First, using a Chernoff bound (see [10]) we have that,

$$Prob\left[X^{m^+(\beta)} \leqslant \frac{1}{2}E[X^{m^+(\beta)}]\right] \leqslant \mathrm{e}^{-E[X^{m^+(\beta)}]/8}.$$

We now define $m^+(\beta)$ to be $m^+(\beta) = 32\ell R_{\beta,t}(\beta/\ell)^{1/2R_{\beta,t}}$. For our choice of $m^+(\beta)$ we have $E[X^{m^+(\beta)}] \geqslant 2\beta$, and hence

$$Prob[X^{m^+(\beta)} \leqslant \beta] \leqslant e^{-\beta/4} = q^+(\beta).$$

For the analysis of the algorithm, we view the algorithm as running in phases so long as $\beta \leqslant \gamma/4$. There will be $k = \log(\gamma/2\alpha)$ phases, and in each phase we start with $\beta = 2^j\alpha$ initial replicas, for $0 \leqslant j < k$. The jth phase runs for $m^+(2^j\alpha)/(F^{\mathrm{out}}2^j\alpha)$ rounds. We say that a phase is "good" if by the end of the phase the number of active replicas has at least doubled. The probability that some phase is not good is bounded by,

$$\sum_{j=0}^{k-1} q^+(2^j\alpha) = \left(\sum_{j=0}^{k-1} \mathrm{e}^{-2^j\alpha/4}\right) \leqslant 2e^{-\alpha/4} \leqslant 1/2,$$

for $\alpha \geqslant 6$. Assuming that all the phases are good, at the end half of the replicas are active.

The number of rounds until half the system is active is at most,

$$\sum_{j=0}^{k-1} \frac{m^+(2^j\alpha)}{F^{\text{out}}2^j\alpha} \leqslant \sum_{j=0}^{k-1} \frac{32\ell R_{2^j\alpha,t}(2^j\alpha/\ell)^{1/(2R_{2^j\alpha,t})}}{F^{\text{out}}2^j\alpha}$$

$$\leqslant \frac{32\ell R_{\alpha,t}}{F^{\text{out}}\alpha}\sum_{j=0}^{k-1}\frac{(2^j\alpha/\ell)^{1/(2R_{\alpha,t})}}{2^j}$$

$$= \mathrm{O}\left(\frac{R_{\alpha,t}}{F^{\text{out}}}\left(\frac{\ell}{\alpha}\right)^{1-1/2R_{\alpha,t}}\right),$$

where we used here the fact that $R_{\beta,t}$ is a decreasing function in β.

We now reach the last stage of the algorithm, when $\beta \geqslant \gamma/2 \geqslant \ell/2$. Unfortunately, there are too few passive replicas to use the analysis above for $m^+(\beta)$. We therefore employ a different technique here.

We give an upper bound on the expected number of rounds for completion at the last stage. Fix any replica p, and let V_i be the number of new updates in round i that p receives. Since $t \leqslant \gamma/4$, we have $\beta - t \geqslant \gamma/4 \geqslant \ell/4$, and so:

$$E[V_i] \geqslant (\beta - t)\frac{F^{\text{out}}}{4\ell} \geqslant \frac{F^{\text{out}}}{16}.$$

Let V^r denote the number of new updates received by p in r rounds, hence $V^r = \sum_{i=1}^{r} V_i$. By linearity of expectation, $E[V^r] \geqslant rF^{\text{out}}/16$. Using a Chernoff bound we have,

$$Prob[V^r < rF^{\text{out}}/32] \leqslant \mathrm{e}^{-rF^{\text{out}}/256}.$$

Let $r^+ = (32t + 2 \times 256\log(\ell))/F^{\text{out}}$. The probability that $V^{r^+} < t$ is at most $1/\ell^2$. The probability that some replica at the root receives $<t$ new updates in r^+ rounds is thus $<1/\ell$, and so in an expected $\mathrm{O}((t + \log(\ell))/F^{\text{out}})$ rounds the algorithm terminates.

Putting the two bounds together, we have an expected $\mathrm{O}((R_{\alpha,t}/F^{\text{out}})(\ell/\alpha)^{(1-1/2R_{\alpha,t})} + (t + \log(\ell))/F^{\text{out}})$ number of rounds. □

When $\ell = n$ the tree degenerates to one node containing the entire system, and the ℓ-Tree method reduces to a random propagation scheme: Simply, each replica, at each round, chooses F^{out} replicas uniformly and at random from all replicas and sends messages to them. Lemma 5.4 above shows that the delay of random propagation in a Byzantine setting is:

Corollary 5.5. *The delay of random propagation is* $\mathrm{O}((R_{\alpha,t}/F^{\text{out}})(n/\alpha)^{(1-1/2R_{\alpha,t})} + (t + \log(n))/F^{\text{out}})$ *for* $2 < t \leqslant n/4$.

The next step of the analysis is to bound how much time it takes from when all the correct replicas in some node become active until its child becomes active.

We will not be interested in the expected time, but rather focus on the time until there is at least a constant probability that the child is active, and show a bound of $\mathrm{O}((t+\log\ell)/F^{\mathrm{out}})$ rounds. The proof is similar to the last part of the proof of Lemma 5.4.

Lemma 5.6. *With probability $\frac{1}{2}$, the number of rounds from the time all the correct replicas in some node become active and until all the correct replicas in its child node are active is* $\mathrm{O}((t+\log\ell)/F^{\mathrm{out}})$.

Proof. Given that $\ell - t$ correct replicas in the parent node are active, each replica in the child node (which is not already active) has an expectation of receiving at least $(\ell-2t)F^{\mathrm{out}}/4\ell \geqslant F^{\mathrm{out}}/8$ updates from new replicas in every round. Using a Chernoff bound, this implies that in r rounds, with probability at least $\mathrm{e}^{-F^{\mathrm{out}}r/64}$, we have $F^{\mathrm{out}}r/16$ updates. This implies that for $r=(16t+2\times 64\log\ell)/F^{\mathrm{out}}$, with probability at most $1/\ell^2$, we have less than t updates. Summing over all replicas in the node gives completes the proof of the lemma. □

We use the result of Lemma 5.6 to bound the total propagation time from the root node to the leaves.

Lemma 5.7. *The expected number of rounds from when all the correct replicas in the root node are active and until all correct replicas are active is* $\mathrm{O}((t+\log\ell)\log(n/\ell)/F^{\mathrm{out}})$, *when* $\ell<n$.

Proof. Each leaf node has $\log(n/\ell)$ nodes on the path leading from the root to it. Partition the rounds into meta-rounds, each containing $\mathrm{O}((t+\log\ell)/F^{\mathrm{out}})$ rounds. For each meta-round there is a probability of at least $\frac{1}{2}$ that another node on the path becomes active. This implies that in k meta-rounds, we have an expected number of $k/2$ active nodes on the path. Therefore, the probability that we have less than $k/4$ is at most $\mathrm{e}^{-k/16}$. We have $\log(n/\ell)$ nodes on the path, this gives the constraint that $k\geqslant 4\log(n/\ell)$. In addition we would like the probability that there exists a leaf node that does not become active to be less than $(\ell/n)^2$, which holds for $k\geqslant 32\log(n/\ell)$. Consider $k=32\log(n/\ell)$ meta rounds. Since there are at most n/ℓ leaves in the tree, with probability at least $1-\ell/n\geqslant 2/3$ the number of meta-rounds is at most $k=\mathrm{O}(\log(n/\ell))$. Thus, the delay is $\mathrm{O}((t+\log\ell)\log(n/\ell)/F^{\mathrm{out}})$. □

The main theorem of this section immediately follows from Lemmas 5.4 and 5.7:

Theorem 5.8. *The delay of the ℓ-Tree propagation method is* $\mathrm{O}((R_{\alpha,t}/F^{\mathrm{out}})(\ell/\alpha)^{(1-1/2R_{\alpha,t})}+\log(\ell)/F^{\mathrm{out}}+(t+\log\ell)\log(n/\ell)/F^{\mathrm{out}})$.

We note that in the analysis above we did not attempt to optimize for the best constants. In fact, much of the constant factor in the Tree diffusion delay can be eliminated if we modify the algorithm to propagate messages deterministically down the tree (but continue selecting targets at random from the root node).

Table 1
Properties of ℓ-Tree

Method	Fan-in	Delay
ℓ-Tree	$O(nF^{\mathrm{out}}/\ell + \log n)$	$O((t/F^{\mathrm{out}})(\ell/\alpha)^{(1-1/3t)} + \log(\ell)/F^{\mathrm{out}} + (t/F^{\mathrm{out}})\log(n/\ell))$
n-Tree (random)	$O(F^{\mathrm{out}} + \log n)$	$O((t/F^{\mathrm{out}})(n/\alpha)^{(1-1/3t)} + \log(n)/F^{\mathrm{out}})$
$4t$-Tree	$O(nF^{\mathrm{out}}/t + \log n)$	$O((t\log(n/t))/F^{\mathrm{out}})$

6. Discussion

Our results for the ℓ-Tree algorithms are summarized in Table 1. The table highlights the two special cases we have already discussed, namely, n-Tree (random propagation) and $4t$-Tree.

Using the fan-in/delay bound of Theorem 4.2, we now examine our results. The n-Tree has $O(\log n)$-amortized fan-in of $O(F^{\mathrm{out}})$, yielding a product of delay and amortized fan-in of $O(t(n/\alpha)^{(1-1/3t)} + \log(n))$ when $\alpha \geqslant 2t$. This is slightly inferior to the lower bound in the range of t for which the lower bound applies. The case $\ell = 4t$ has fan-in (and amortized fan-in) of $O(nF^{\mathrm{out}}/t)$ and delay $O((\log(\alpha))/F^{\mathrm{out}} + (t/F^{\mathrm{out}})\log(n/t))$ if $\alpha \geqslant 2t$. So, their product is $O((n\log(\alpha))/t + n\log(n/t))$, which again is inferior to the lower bound of $\Omega(tn/\alpha)$ since $t/\alpha \leqslant 1$. However, recall from Theorem 4.1 that the delay is always $\Omega((t/F^{\mathrm{out}})\log(n/\alpha))$, and so for the fan-in of $O(nF^{\mathrm{out}}/t)$ it is impossible to achieve optimal delay/fan-in tradeoff. In the general ℓ-Tree method, putting $\ell \geqslant \alpha\log(n/\alpha)$, the ℓ-Tree algorithm exhibits a fan-in/delay product of at most $O(tn/\alpha)$, which is optimal. If $\ell < \alpha\log(n/\alpha)$, the product is within a logarithmic factor from optimal. Hence, ℓ-Tree propagation provides a spectrum of protocols that have optimal delay/fan-in tradeoff in our model to within a logarithmic factor.

7. Expiring updates

Our delay analysis plays a crucial role in *expiring* updates, and hence, in keeping the storage and communication costs of diffusion bounded. That is, in practice a replica must eventually discard each update after propagating it for some number of rounds. The number of rounds to propagate an update is determined by the prediction provided by our analysis for completion of diffusion. More specifically, say that the predicted delay of the diffusion method for a particular system is d. When a replica receives an update directly from the source, it sets a *time-to-live* (TTL) field for the propagated update to $d + \varepsilon$, where ε is a parameter of the system. In each round, it decrements TTL and, so long as it is still positive, sends the update to a target chosen according to the diffusion method. Finally, when TTL reaches zero, the update is "expired": the replica removes it from its collection of updates to propagate, thereby releasing the storage it consumed.

When a replica becomes active for an update through propagation, i.e., after it receives copies of the update from t distinct replicas, it similarly must set a TTL for propagating this update. One alternative would be for this replica to set its TTL for propagating this update to the minimum TTL among all copies received so far, which would make sense in a system without failures and with predictable message delays. However, this approach is vulnerable to faulty replicas, in the sense that a faulty replica could send a copy with a low TTL to terminate the propagation of the update prematurely. We thus suggest a conservative approach to calculate the TTL, namely, the replica sets the TTL for this update to the maximum of all t received TTLs minus one. In this case, faulty replicas can cause updates to linger in the system longer than necessary by sending copies with inflated TTLs. However, they cannot cause the propagation of an update to end prematurely.

References

[1] K.P. Birman, M. Hayden, O. Ozkasap, Z. Xiao, M. Budio, Y. Minsky, Bimodal multicast, ACM Trans. Comput. Systems 17 (2) (1999) 41–88.

[2] A.D. Birrell, R. Levin, R.M. Needham, M.D. Schroeder, Grapevine, An exercise in distributed computing, Commun. ACM 25 (4) (1982) 260–274.

[3] G. Bracha, S. Toueg, Asynchronous consensus and broadcast protocols, J. ACM 32 (4) (1985) 824–840.

[4] M. Castro, B. Liskov, Practical Byzantine fault tolerance, in: Proc. 3rd Symp. on Operating Systems Design and Implementation, 1999.

[5] F. Cristian, H. Aghili, R. Strong, D. Dolev, Atomic broadcast: from simple message diffusion to Byzantine agreement Inform. Comput. 18 (1) (1995) 158–179.

[6] S.E. Deering, Host extensions for IP multicasting, SRI Network Information Center, RFC 1112, 1989.

[7] A. Demers, D. Greene, C. Hauser, W. Irish, J. Larson, S. Shenker, H. Sturgis, D. Swinehart, D. Terry, Epidemic algorithms for replicated database maintenance, in: Proc. 6th ACM Symp. Principles of Distributed Computing, 1987, pp. 1–12.

[8] D. Dolev, R. Strong, Authenticated algorithms for Byzantine agreement, SIAM J. Comput. 12 (4) (1983) 656–666.

[9] W. Hoeffding, Probability inequalities for sums of bounded random variables, J. Amer. Statist. Assoc. 58 (301) (1963) 13–30.

[10] M.J. Kearns, U.V. Vazirani, An Introduction to Computational Learning Theory, MIT Press, Cambridge, MA, 1994.

[11] K.P. Kihlstrom, L.E. Moser, P.M. Melliar-Smith, The SecureRing protocols for securing group communication, in: Proc. 31st IEEE Ann. Hawaii Internat. Conf. on System Sciences, Vol. 3, 1998, pp. 317–326.

[12] L. Lamport, R. Shostak, M. Pease, The Byzantine generals problem, ACM Trans. Programm. Languages Systems 4 (3) (1982) 382–401.

[13] K. Lidl, J. Osborne, J. Malcome, Drinking from the firehose: multicast USENET news, in: Proc. Usenix Winter Conf., 1994, pp. 33–45.

[14] D. Malkhi, Y. Mansour, M. Reiter, On diffusing updates in a Byzantine environment, in: Proc. 18th IEEE Symp. on Reliable Distributed Systems, 1999, pp. 134–143.

[15] D. Malkhi, E. Pavlov, Y. Sella, Optimal unconditional information diffusion, in: Proc. 15th Internat. Symp. on Distributed Computing, October 2001.

[16] D. Malkhi, M. Reiter, A high-throughput secure reliable multicast protocol, J. Comput. Security 5 (1997) 113–127.

[17] D. Malkhi, M. Reiter, Byzantine quorum systems, Distributed Comput. 11 (4) (1998) 203–213.

[18] D. Malkhi, M.K. Reiter, An architecture for survivable coordination in large distributed systems, IEEE Trans. Knowledge Data Eng. 12 (2) (2000) 187–202.

[19] D. Malkhi, M. Reiter, O. Rodeh, Y. Sella, Efficient update diffusion in Byzantine environments, in: Proc. 20th Symp. on Reliable Distributed Systems (SRDS 2001), October 2001.

[20] D. Malkhi, M.K. Reiter, D. Tulone, E. Ziskind, Persistent objects in the Fleet system, in: Proc. 2nd DARPA Inform. Survivability Conf. and Exposition, June 2001.

[21] D. Malkhi, M.K. Reiter, A. Wool, R.N. Wright, Probabilistic quorum systems, Inform. Comput. 170 (2) (2001).

[22] D. Malkhi, Y. Sella, Replication by diffusion in large networks, in: European Research Seminar on Advances in Distributed Systems (Ersads 2001), Bologna, Italy, 2001.

[23] Y. Minsky, Spreading rumors cheaply, quickly, and reliably, Ph.D. Thesis, Department of Computer Science, Cornell University, 2002.

[24] L.E. Moser, P.M. Melliar-Smith, Total ordering algorithms for asynchronous Byzantine systems, in: Proc. 9th Internat. Workshop on Distributed Algorithms, Springer, 1995.

[25] R. Motwani, P. Raghavan, Randomized Algorithms, Cambridge University Press, Cambridge, 1995.

[26] D.C. Oppen, Y.K. Dalal, The Clearinghouse: a decentralized agent for locating named objects in a distributed environment, Xerox Technical Report: OPD-T8103, 1981.

[27] B. Pittel, On spreading a rumor, SIAM J. Appl. Math. 47 (1) (1987) 213–223.

[28] M.K. Reiter, Secure agreement protocols: reliable and atomic group multicast in Rampart, in: Proc. 2nd ACM Conf. on Computer and Communications Security, November 1994, pp. 68–80.

[29] G.J. Simmons, A survey of information authentication, in: Contemporary Cryptology, The Science of Information Integrity, Wiley-IEEE Press, New York, 1999.