Brahms: Byzantine Resilient Random Membership Sampling

Edward Bortnikov* Maxim Gurevich* Idit Keidar* Gabriel Kliot†
Alexander Shraer*

Abstract

We present Brahms, an algorithm for sampling random nodes in a large dynamic system prone to Byzantine failures. Brahms stores small membership views at each node, and yet overcomes Byzantine failures of a linear portion of the system. Brahms is composed of two components. The first one is a Byzantine-resistant gossip-based membership protocol. The second one uses a novel memory-efficient approach for uniform sampling from a possibly biased stream of ids that traverse the node. We evaluate Brahms using rigorous analysis, backed by extensive simulations, which show that our theoretical model captures the protocol's essentials. We show that, with high probability, an attacker cannot create a partition between correct nodes. We further prove that each node's sample converges to a uniform one over time. To our knowledge, no such properties were proven for gossip-based membership in the past.

Keywords: random sampling, gossip, membership, Byzantine failures.

[1]Department of Electrical Engineering, The Technion – Israel Institute of Technology.
Email: {ebortnik@techunix, gmax@techunix, idish@ee, shralex@techunix}.technion.ac.il.
[2]Department of Computer Science, The Technion – Israel Institute of Technology.
Email: gabik@cs.technion.ac.il.

1 Introduction

We consider the problem of sampling a random node (or peer) in a large dynamic system subject to Byzantine (arbitrary) failures. Random node sampling is important for many scalable dynamic applications, including neighbor selection in constructing and maintaining overlay networks [13, 19, 22, 24], selection of communication partners in gossip-based protocols [5, 8, 11], data sampling, and choosing locations for data caching, e.g., in unstructured peer-to-peer networks [21].

Typically, in such applications, each node maintains a set of random node ids that is asymptotically smaller than the system size. This set is called a *local view*. In a dynamic system, where the set of active nodes changes over time (this is called *churn*), the local views must continuously evolve to reflect these changes, adding new active nodes and removing ones that are no longer active. By using small local views, the maintenance overhead is kept small. In the absence of Byzantine failures, small local views can be effectively maintained with gossip-based membership protocols [1, 11, 12, 16, 28], which were proven to have a low probability for partitions, including under churn [1].

Nevertheless, Byzantine failures present a major challenge for small local views. Previous Byzantine-tolerant gossip protocols either considered static settings where the full membership is known to all [9, 20, 26], or maintained (almost) full local views [3, 17], where faulty nodes cannot push correct ones out of the view. In contrast, small local views are susceptible to poisoning with entries (node ids) originating from faulty nodes; this is because the system is dynamic, and therefore nodes inherently must accept new ids and store them in place of old ones in their local views. It is even more challenging to provide *independent uniform samples* in such a setting. Even without Byzantine failures, gossip-based membership only ensures that eventually the *average* representation of nodes in local views is uniform [1, 12, 16], and not that *every node* obtains an independent uniform random sample. Faulty nodes may attempt to skew the system-wide distribution, as well as the individual local view of a given node.

In this paper, we address these challenges. We present Brahms (Section 3), a gossip-based membership service that stores a sub-linear number of ids (e.g., $\Theta(\sqrt[3]{N})$ in a system of size N) at each node, and provides *each node* with membership samples that converge to uniform ones over time. The main ideas behind Brahms are (1) to use gossip-based membership with some extra defenses to make it viable (in the sense that local views are not solely composed of faulty ids) in a Byzantine setting; (2) to recognize that such a solution is bound to produce biased views due to attacks (we precisely quantify the extent of this bias mathematically); and (3) to correct this bias at each node.

To achieve the latter, we introduce *Sampler*, a component that obtains uniform samples out of a data stream in which elements recur with an unknown bias, using min-wise independent permutations [6]. We prove (Section 4) that Sampler obtains independent uniform samples from the biased stream of gossiped node ids. By using such *history samples* of the gossiped ids to update part of the local view, Brahms achieves *self-healing* from partitions that may occur with gossip-based membership. In particular, nodes that have been active for sufficiently long (we quantify how long) cannot be isolated from the rest of the system. The use of history samples is an example of *amplification*, whereby even a small healthy sample of the past can boost the resilience of a constantly evolving view. We note that only a small portion of the view is updated with history samples, e.g., 10%. Therefore, the protocol can still deal effectively with churn.

One of the important contributions of this paper is our mathematical analysis (Section 5), which provides insights to the extent of damage that Byzantine nodes can cause and the effectiveness of various mechanisms for dealing with them. Extensive simulations of Brahms with up to 4000 nodes validate the few simplifying assumptions made in the analysis. We consider two possible goals for an attacker. First, we study attacks that attempt to maximize the representation of faulty ids in local views at any given time. We show that as

long as faulty nodes comprise less than $\frac{1}{3}$ of the system, even without using history samples, the portion of faulty ids in local views is bounded by a constant smaller than one. (Recall that the over-representation of faulty ids is later fixed by Sampler; the upper bound on faulty ids in local views ensures Sampler has good ids to work with). If the adversary gains control of additional nodes after uniform samples have already been obtained, then Brahms can resist *any* ratio of faulty nodes.

Second, we consider an attacker that aims to partition the network. The easiest way to do so is by attempting to isolate one node from the rest. Clearly, once a node has obtained uniform samples of correct nodes, it can no longer be isolated. We therefore study an attack launched on a new node that joins the system when its samples are still empty, and when it does not yet appear in views or samples of other nodes. We further assume that such a *targeted* attack on the new node occurs in tandem with an attack on the entire system, as described above. The key to proving that Brahms prevents, w.h.p., an attacked node's isolation is in comparing how long it takes for two competing processes to complete: on the one hand, we provide a lower bound on the expected time to poison the entire view of the attacked node, assuming there are no history samples at all. On the other hand, we provide an uppper bound on how fast history samples are expected to converge, under the same attack. Whenever the former exceeds the latter, the attacked node is expected to become immune to isolation before it is isolated. We prove that with appropriate parameter settings, this is indeed the case.

Finally, we simulate the complete system (Section 6), and measure Brahms's resilience to the combination of both attacks. Our results show that, indeed, Brahms prevents the isolation of attacked nodes, its views never partition, and the membership samples converge to perfectly random ones over time.

Related Work. We are not familiar with prior work dealing with random node sampling in a Byzantine setting. Previous Byzantine-tolerant membership services maintained full local views [17, 3] rather than partial samples. Previous work on gossip-based partial views [1, 11, 12, 16, 28], and on near-uniform node sampling using random walks [13, 19, 23, 4] or DHT overlays [18] was limited to benign settings.

One application of Brahms is Byzantine-tolerant overlay construction. Brahms's sampling allows each node to connect with some random correct nodes, thus constructing an overlay in which the sub-graph of correct nodes is connected. Several recent works, e.g., [27, 7, 2], have focused explicitly on securing overlays, mostly structured ones, attempting to ensure that all correct nodes may communicate with each other using the overlay, i.e., to prevent the *eclipse attack* [27], where routing tables of correct nodes are gradually poisoned with links to adversarial nodes. These works have a different focus than ours, since their goal is to construct (structured) overlay networks, whereas we present a general sampling technique, one application of which is building Byzantine-resilient unstructured overlays.

2 Model and Required Properties

2.1 System Model

We consider a dynamic set of nodes, each of which can be either *active* or *passive* at any given time. Each node is identified by a unique id, chosen when the node becomes active for the first time. The set of active nodes at time t is denoted $\mathcal{A}(t)$. Active nodes can communicate through a fully connected network with reliable links. For simplicity of the analysis, we assume a synchronous model with a discrete global clock, zero processing times, and message latencies of a single time unit.

Some of the active nodes are *correct*, and the rest are *faulty*. Faulty modes can exhibit arbitrary behavior (Byzantine faults). The subset of correct nodes in $\mathcal{A}(t)$ is denoted $\mathcal{C}(t)$. Nodes can determine the source of every message and cannot intercept messages addressed to other nodes (the standard "unauthenticated" Byzantine model).In static systems (without churn), it is common to require that faulty nodes comprise less

than some fraction $f < 1$ of the nodes. In a dynamic setting, we require that the number of faulty nodes at all times is limited by a constant fraction f of the minimal number of active nodes, i.e., $|\bigcup_t (\mathcal{A}(t) \setminus \mathcal{C}(t))| \leq f \cdot \min_t |\mathcal{A}(t)|$. While this assumption rules out massive Sybil attacks [10] (by bounding the number of faulty node ids), it is weaker than assuming a certification authority [17], e.g., nodes can use historic information for choosing new active ids.

We assume a mechanism that makes it costly for nodes to send designated messages, which we call *limited* messages, thereby limiting their sending rate. This mechanism can be implemented in different ways, e.g., computational challenges like Merkle's puzzles [25], virtual currency, etc. We assume that the system-wide fraction of limited messages that all faulty nodes can jointly send in a single time unit is at most p, for some $p < 1$. We also assume that the faulty nodes choose the destinations of all limited messages in advance (i.e., they do not adapt their transmissions to information learned during the run).

2.2 Membership Sampling Specification

At all times t, Brahms provides two tuples at every active correct node u: a *neighbor list* $\mathcal{N}_u(t)$ used for communication, and a *sample list* $\mathcal{S}_u(t)$. These lists may contain duplicates, and some entries in $\mathcal{S}_u(t)$ may be non-defined (denoted $\perp$). We denote the i'th element in the neighbor list and the sample list at time t by $\mathcal{N}_u^i(t)$ and $\mathcal{S}_u^i(t)$, respectively. Every correct node has a limited local storage, asymptotically smaller than the maximal size of the active node set (i.e., both $\mathcal{N}_u(t)$ and $\mathcal{S}_u(t)$ are asymptotically smaller than $\mathcal{A}(t)$).

First, *we require the overlay induced by $\mathcal{N}$ to remain connected w.h.p.* Formally, we define a dynamic directed *overlay graph*, which captures the knowledge of correct nodes about each other at each time t:

$$\mathcal{N}(t) \triangleq \{\mathcal{C}(t), \bigcup_{u \in \mathcal{C}(t)} \{(u,v) | v \in \mathcal{N}_u(t) \cap \mathcal{C}(t)\}\}.$$

Requirement 1 With high probability, $\mathcal{N}(t)$ remains weakly connected at all t.

Next, *we require $\mathcal{S}$ to converge to a uniform sample of the connected overlay*. However, when the set of active nodes is constantly changing, the notion of a uniform distribution over it is meaningless. Hence, like previous specifications [1, 12, 16], we consider the system's properties after a point T_0 when the churn of *correct* nodes ceases (i.e., $\mathcal{C}(t) = \mathcal{C}(T_0)$ for all $t \geq T_0$). We are interested in *eventual independent uniform* sampling from $\mathcal{C}(T_0)$. Note that we cannot require the same from the set of faulty nodes, since their behavior is arbitrary. However, we require that (1) the probability of a sample being faulty does not exceed the maximal fraction of faulty ids in $\mathcal{A}(t)$ after T_0, and (2) the probability of a sample being each specific correct id is eventually between $\frac{1}{|\mathcal{C}(t)|}$ and $\frac{1}{|\mathcal{A}(t)|}$. Formally,

Requirement 2 *If $\mathcal{N}(t)$ is weakly connected for all $t \geq T_1 \geq T_0$, then for all $u, v \in \mathcal{C}(T_0)$, all samples i, and all $\varepsilon > 0$, there exists $T_\varepsilon \geq T_1$ such that for all $t \geq T_\varepsilon$*

$$\frac{1}{\max_{T \geq T_0} |\mathcal{A}(T)|} - \varepsilon \leq Pr[\mathcal{S}_u^i(t) = v] \leq \frac{1}{|\mathcal{C}(T_0)|} + \varepsilon.$$

In other words,

$$\frac{1-f}{|\mathcal{C}(T_0)|} - \varepsilon \leq Pr[\mathcal{S}_u^i(t) = v] \leq \frac{1}{|\mathcal{C}(T_0)|} + \varepsilon.$$

```
1: function Sampler.init()
2:     h ← randomPRF(); q ← ⊥
3: function Sampler.next(elem)
4:     if q = ⊥ ∨ h(elem) < h(q) then
5:         q ← elem
6: function Sampler.sample()
7:     return q
```

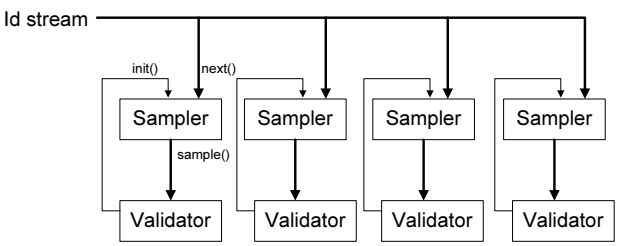


Figure 1: **Uniform sampling from an id stream in Brahms. (a) Sampler's pseudo-code. (b) Sampling and validation of ℓ_2 ids.**

3 Brahms

Brahms has two components. The local *sampling* component maintains a *sample list* $\mathcal{S}$ – a tuple of uniform samples from the set of ids that traverse the node (Section 3.1). The *gossip* component is a distributed protocol that spreads locally known ids across the network (Section 3.2), and maintains a dynamic *view* $\mathcal{V}$. Each node has some initial $\mathcal{V}$ (e.g., received from some bootstrap server or peer node). The concatenation of $\mathcal{V}$ and $\mathcal{S}$ (denoted $\mathcal{V} \circ \mathcal{S}$) is the node's *neighbor list* $\mathcal{N}$.

3.1 Sampling

Sampler is a building block for uniform sampling of unique elements from a data stream. The input stream may be biased, that is, some values may appear in it more than others. Sampler accepts one element at a time as input, produces one output, and stores a single element at any time. The output is a uniform random choice of one of the unique inputs witnessed thus far.

Sampler uses *min-wise independent* permutations [6]. A family of permutations $\mathcal{H}$ over a range $[1 \ldots |U|]$ is min-wise independent if for any set $X \subset [1 \ldots |U|]$ and any $x \in X$, if h is chosen at random from $\mathcal{H}$, then $\Pr(\min\{h(X)\} = h(x)) = \frac{1}{|X|}$. That is, all the elements of any fixed set X have an equal chance to have the minimum image under h. Pseudo-random (hash) functions (e.g., [14]) are considered an excellent practical approximation of min-wise independent permutations, provided that $|U|$ is large, e.g., 2^{160}.

Sampler (Figure 1(a)) selects a random min-wise independent function h upon initialization, and applies it to all input values (`next()` function). The input with the smallest image value encountered thus far becomes the output returned by the `sample()` function. The property of uniform sampling from the set of unique observed ids follows directly from the definition of a min-wise independent permutation family.

Brahms maintains a tuple of ℓ_2 sampled elements in a vector of ℓ_2 Sampler blocks (Figure 1(b)), which select hashes independently. The same id stream is input to all Samplers. Sampled ids are periodically probed (e.g., using pings), and a Sampler that holds an inactive node is invalidated (re-initialized).

3.2 Gossip

Brahms's view is maintained by a gossip protocol (Figure 2). By slight abuse of notation, we denote both the vector of samplers and their outputs (the sample list) by $\mathcal{S}$. Brahms executes in (unsynchronized) rounds. It uses two means for propagation: (1) *push* – sending the node's id to some other node, and (2) *pull* – retrieving the view from another node. These operations serve two different purposes: pushes are required to reinforce knowledge about nodes that are under-represented in other nodes' views (e.g., newborn nodes), whereas pulls are needed to spread existing knowledge within the network [1]. A combination of pulls and

```
1: 𝒱 : tuple[ℓ1] of Id
2: 𝒮 : tuple[ℓ2] of Sampler
3: 𝒩 ≜ 𝒱 ∘ 𝒮
4: Initialization(𝒱0):
5:    𝒱 ← 𝒱0
6:    for all 1 ≤ i ≤ ℓ2 do
7:       𝒮[i].init()
8:    updateSample(𝒱0)
9: {Stale sample invalidation}
10: periodically do
11:    for all 1 ≤ i ≤ ℓ2 do
12:       if probe(𝒮[i].sample()) fails then
13:          𝒮[i].init()
14: {Auxiliary functions}
15: function updateSample(𝒱)
16:    for all id ∈ 𝒱, 1 ≤ i ≤ ℓ2 do
17:       𝒮[i].next(id)
18: function rand(𝒱, n)
19:    return n random choices from 𝒱
20: {Gossip}
21: while true do
22:    𝒱push ← 𝒱pull ← ∅
23:    for all 1 ≤ i ≤ αℓ1 do
24:       {Limited push}
25:       send_lim ⟨"push_request"⟩ to rand(𝒱, 1)
26:    for all 1 ≤ i ≤ βℓ1 do
27:       send ⟨"pull_request"⟩ to rand(𝒱, 1)
28:    wait(1)
29:    for all received ⟨"push_request"⟩ from id do
30:       𝒱push ← 𝒱push ∘ {id}
31:    for all received ⟨"pull_request"⟩ from id do
32:       send ⟨"pull_reply", 𝒱⟩ to id
33:    for all received ⟨"pull_reply", 𝒱′⟩ from id do
34:       if I sent the request, and this is the first reply then
35:          𝒱pull ← 𝒱pull ∘ 𝒱′
36:    if (|𝒱push| ≤ αℓ1 ∧ 𝒱push ≠ ∅ ∧ 𝒱pull ≠ ∅) then
37:       𝒱 ← rand(𝒱push, αℓ1) ∘ rand(𝒱pull, βℓ1) ∘ rand(𝒮, γℓ1)
38:    updateSample(𝒱push ∘ 𝒱pull)
```

Figure 2: **The pseudo-code of Brahms.**

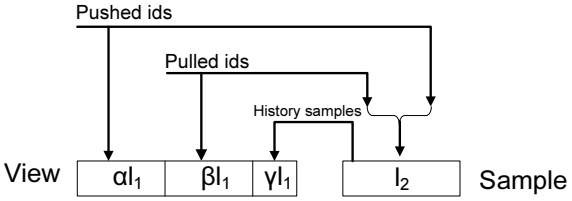


Figure 3: **View re-computation in Brahms.**

pushes is required because the representation of ids propagated solely by pulls decays over time, whereas the representation of push-propagated ids increases.

Brahms uses parameters $\alpha > 0$, $\beta > 0$ and $\gamma > 0$ that satisfy $\alpha + \beta + \gamma = 1$. In a single round, a correct node issues $\alpha\ell_1$ push requests and $\beta\ell_1$ pull requests to destinations randomly selected from its view (possibly with repetitions). At the end of each round, $\mathcal{V}$ and $\mathcal{S}$ are updated with fresh ids. While all received ids are streamed to $\mathcal{S}$ (Figure 1, Line 38), re-computing $\mathcal{V}$ requires extra care, to protect against poisoning of the views with faulty ids. Brahms offers a set of techniques sto mitigate this problem.

Limited pushes. Since pushes arrive unsolicited, an adversary with an unlimited capacity could swamp the system with push requests. Then, correct ids would be propagated mainly through pulls, and their representation would decay exponentially [1]. Brahms employs limited push messages, hence the fraction of faulty pushes does not exceed p.

Attack detection and blocking. While using limited pushes prevents a simultaneous attack on all correct nodes, it provides no solace against an adversary that floods a specific node. Brahms protects against this *targeted attack* by blocking the update of $\mathcal{V}$. Namely, if more than the expected $\alpha\ell_1$ pushes are received, it does not update $\mathcal{V}$. Although this policy slows down progress, its expected impact in the absence of attacks is bounded (nodes recompute $\mathcal{V}$ in most rounds). Thanks to limited pushes, some nodes make progress even under an attack (Line 36).

Controlling the contribution of pushes vs pulls. As most correct nodes do not suffer from targeted attacks (due to limited pushes), their views are threatened by pulls from neighbors more than by adversarial pushes.

This is because whereas all pushes from correct nodes are correct, a pull from a random correct node may contribute some faulty ids. Hence, the contribution of pushes and pulls to $\mathcal{V}$ must be balanced: pushes must be constrained to protect the targeted nodes, while pulls must be constrained to protect the rest. Brahms updates $\mathcal{V}$ with randomly chosen $\alpha\ell_1$ pushed ids and $\beta\ell_1$ pulled ids (Line 37).

History samples. The attack detection and blocking technique can slowdown a targeted attack, but not prevent it completely. Note that if the adversary succeeds to increase its representation in a victim's view through targeted pushes, it subsequently causes this victim to pull more data from faulty nodes. As the attacked node's view deteriorates, it sends fewer pushes to correct nodes, causing its system-wide representation to decrease. It then receives fewer correct pushes, opening the door for more faulty pushes[1]. Brahms overcomes such attacks using a self-healing mechanism, whereby a portion (γ) of $\mathcal{V}$ reflects the *history*, i.e., previously observed ids (Line 37). A direct use of history does not help since the latter may also be biased. Therefore, we use a feedback from $\mathcal{S}$ to obtain unbiased history samples. Once some correct id becomes the attacked node's permanent sample (or the node's id becomes a permanent sample of some other correct node), the threat of isolation is eliminated. Figure 3 illustrates the entire view re-computation procedure.

Brahms's parameters entail a tradeoff between performance in a benign setting and resilience against Byzantine attacks. For example, γ must not be too large since the algorithm needs to deal with churn; on the other hand, it must not be too small to make the feedback effective. We show (Section 6) that $\gamma = 0.1$ is enough for protecting $\mathcal{V}$ from partitions. The choice of ℓ_1 and ℓ_2 is crucial for guaranteeing that a targeted attack can be contained until the attacked node's sample stabilizes. For example, $\ell_1, \ell_2 = \Theta(\sqrt[3]{|\mathcal{C}(T_0)|})$ suffice to protect even nodes that are attacked immediately upon joining the system (Section 5.2).

4 Analysis - Sampling

In this section we analyze the properties of $\mathcal{S}_u$ of a correct node u. Recall that each Sampler R employs a min-wise independent permutation $R.h$, chosen independently at random. Let $R(t)$ be the output of R at time t. We define the *perfect* id corresponding to R, $V_R \in \mathcal{A}(T_0)$, to be the id with the minimal value of $R.h$ in $\mathcal{A}(T_0)$ (we neglect collisions for the sake of the definition). Note that V_R can be either a correct or a faulty id. In Section 4.1 we show that the subset of correct ids in $\mathcal{S}_u$ eventually converges to a uniform random sample from $\mathcal{C}(T_0)$. In Section 4.2 we analyze how fast a node obtains at least one correct perfect sample, as needed for self-healing. Section 4.3 discusses scalability, namely, how to choose view sizes that ensure a constant convergence time, independent of system size.

4.1 Eventual Convergence to Uniform Sample

Consider Sampler R. Given that V_R is correct, R samples correct ids uniformly at random. If V_R is faulty, it may refrain from answering pings and become invalidated instead of remaining in the sample. However, since faulty nodes do not adapt to the correct nodes' random choices, we assume that such an invalidation is not timed in order to capture any particular correct id into R. We therefore assume that each correct id has an equal probability for taking the place of an invalidated faulty node. The following theorem shows that $\mathcal{S}_u$ satisfies Requirement 2 of the membership sampling specification (see Section 2.2).

[1] This avalanche process can be started, e.g., by opportunistically sending the target a slightly higher number of pushes than expected. Since correct pushes are random, a round in which sufficiently few correct pushes arrive, such that Brahms does not detect an attack, happens soon w.h.p.

Theorem 4.1 *If $\mathcal{N}(t)$ remains weakly connected for each $t \geq T_1$, for some $T_1 \geq T_0$, then, for all $v \in \mathcal{C}(T_0)$, and $\varepsilon > 0$, there exists $T_\varepsilon \geq T_1$ such that for all $t \geq T_\varepsilon$*

$$\frac{1-f}{|\mathcal{C}(T_0)|} - \varepsilon \leq \Pr(R(t) = v) \leq \frac{1}{|\mathcal{C}(T_0)|} + \varepsilon.$$

Proof idea (see Appendix A.1). The key to the theorem is to show that whenever $\mathcal{N}(t)$ remains weakly connected, the id of each correct node eventually reaches every other correct node w.h.p. This is because the id has a non-zero probability to traverse a path to every correct node in the system. Thus, each Sampler will eventually settle on its perfect id, provided that its perfect id is correct. Therefore, $\Pr(R(t) = V_R | V_R \in \mathcal{C}(T_0)) \rightarrow_{t\rightarrow\infty} 1$. Since the probability for V_R to be faulty is at most f, $\Pr(R(t) = V_R)$ approaches the range $[1-f, 1]$. The theorem follows since $\forall v \in \mathcal{C}(T_0), \Pr(R(t) = v | V_R \in \mathcal{C}(T_0)) = \frac{1}{|\mathcal{C}(T_0)|}$, and since we assume that when V_R is faulty, $0 \leq \Pr(R(t) = v | V_R \notin \mathcal{C}(T_0)) \leq \frac{1}{|\mathcal{C}(T_0)|}$.

The next lemma discusses the convergence rate of samples.

Lemma 4.2 *If no invalidations happen, for each correct node u, the expected fraction of Samplers that output their perfect id grows linearly with the fraction of unique ids from $\mathcal{A}(T_0)$ observed by u.*

Proof : Let $D(t) \subseteq \mathcal{A}(T_0)$ be the set of ids observed by u until time t. Then, for each sampler R, $\Pr(V_R \in D(t)) = \frac{|D(t)|}{|\mathcal{A}(T_0)|}$. Since for each R such that $V_R \in D(t)$, $R(t') = V_R$ for $t' \geq t$, the lemma follows. □

4.2 Convergence to First Perfect Sample

Here we analyze how many ids have to be observed by a correct node, u, in order to guarantee, w.h.p., that its $\mathcal{S}_u$ contains *at least one* perfect id of an active correct node. This provides an upper bound on the time it takes $\mathcal{S}_u$ ensure self-healing and prevent u's isolation. We assume that u joins the system at time T_0, with an empty sample. Let $\Lambda(t)$ be the number of correct ids observed by u from time T_0 to t. Our analysis depends on the number of unique ids observed by u, rather than directly on Λ. Obviously, one can expect the observed stream to include many repetitions, as it is unrealistic to expect our gossip protocol to produce independent uniform random samples (cf. [16]). Indeed, achieving this property is the goal of Sampler. In order to capture the bias in Λ, we define a *stream deficiency factor*, $0 \leq \rho \leq 1$, so that a stream of length $\Lambda(t)$ produced by our gossip mechanism is roughly equivalent, for the purposes of Sampler, to a stream of length $\rho\Lambda(t)$ in which correct ids are independent and distributed uniformly at random. This is akin to the clustering coefficient of gossip-based overlays [16]. We empirically measured ρ to be about 0.4 with our gossip protocol (see Section 5.2).

We define the *perfect sample probability* $PSP_u(t)$ as the probability that $\mathcal{S}_u(t)$ contains at least one correct perfect id. The convergence rate of PSP is captured by the following:

Lemma 4.3 *Let u be a random correct node. Then, for $t > T_0$, $PSP_u(t) \geq 1 - ((1-f)e^{-\frac{\rho\Lambda(t)}{|\mathcal{C}(T_0)|}} + f)^{\ell_2}$.*

Proof idea (see Appendix A.2). A Sampler outputs a correct perfect id if (1) its perfect id is correct, and (2) this id is observed by the Sampler in the stream. PSP is the probability that at least one of ℓ_2 Samplers outputs a correct perfect id.

Figure 4.2 illustrates the dependence of PSP on the stream size $\Lambda(t)$ and on ℓ_2. When the sample size is $40 = 4\sqrt[3]{|\mathcal{A}(T_0)|}$, and the portion of unique ids in the stream is $\rho = 0.4$, a perfect sample is obtained, w.h.p., after 300 ids traverse the node.

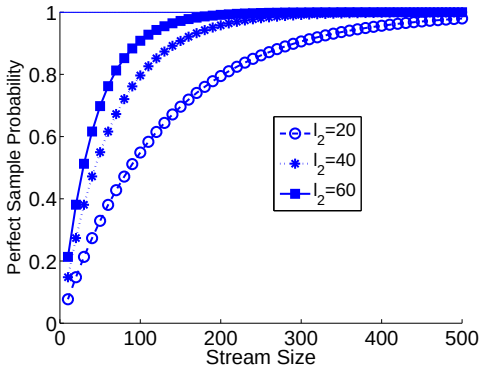


Figure 4: **Growth of the probability to observe at least one correct perfect sample (Perfect Sample Probability - PSP) with the stream size, for** 1000 **nodes,** $f = 0.2$**, and** $\rho = 0.4$**.**

4.3 Scalability

From Lemma 4.3 we see that PSP depends on Λ and ℓ_2. To get a higher PSP, we can increase either one. While increasing Λ is achieved by increasing ℓ_1, and consequently the network traffic, increasing ℓ_2 has only a memory cost. We now study the asymptotic behavior of $PSP_u(t)$ as the number of the nodes, N, increases. When a node has ℓ_2 Samplers, $\Omega(\ell_2)$ of them have correct V_R w.h.p. Therefore, w.h.p., $PSP_u(t) \geq \Omega(1 - (e^{-\frac{\rho\Lambda(t)}{N}})^{\ell_2}) = \Omega(1 - e^{-\frac{\rho\Lambda(t)\ell_2}{N}})$. For a constant t, $\Lambda(t) = \Omega(\ell_1^2)$ since there are $\Omega(\ell_1)$ pulls, obtaining $\Omega(\ell_1)$ ids each. Thus, $PSP_u(t) \geq \Omega(1 - e^{-\frac{\ell_1^2 \cdot \ell_2}{N}})$. For scalability, it is important that for a given t, $PSP_u(t)$ will be bounded by a constant independent of the system size. This condition is satisfied when $\ell_1^2 \cdot \ell_2 = \Omega(N)$, e.g., when $\ell_2 = \ell_1 = \Omega(\sqrt[3]{N})$, or $\ell_1 = \Omega(\sqrt[4]{N})$ and $\ell_2 = \Omega(\sqrt[2]{N})$. To reduce network traffic at the cost of a higher memory consumption, one can set $\ell_1 = \Omega(\log N)$ and $\ell_2 = \Omega(\frac{N}{\log^2 N})$.

5 Evaluation – Overlay Connectivity

We prove that Brahms, with appropriate parameter settings, maintains overlay connectivity despite adversary attacks. Our main methodology is mathematical analysis, which, like previous studies [1], makes some simplifying assumptions. The theoretical results are validated through extensive simulations.

Definitions. We study time-varying random variables, listed in Table 1. A local variable at a specific correct node u is subscripted by $_u$. When used without subscript, a variable corresponds to a random correct node. Correct (resp., faulty) ids propagated through pushes and pulls are denoted g (green) (resp., r (red)). We define $\mathcal{V}(t)$, a subgraph of $\mathcal{N}(t)$ induced by $\mathcal{V}$ of correct nodes: (edges induced by $\mathcal{S}$ are omitted):

$$\mathcal{V}(t) \triangleq \{\mathcal{C}(t), \bigcup_{u \in \mathcal{C}(t)} \{(u, v) | v \in \mathcal{V}_u(t) \cap \mathcal{C}(t)\}\}.$$

For a node u, the number of instances of u in views of correct nodes is called its *in-degree*, and the number of correct ids in $\mathcal{V}_u$ is called its *out-degree*. The *degree* of u is the sum of its in-degree and out-degree.

Assumptions. Brahms's resilience depends on the distribution of in-degrees and out-degrees in $\mathcal{V}(t)$. We assume a necessary condition for initial connectivity, namely, that the view of every joining correct node contains some correct ids (the ratio of faulty ids in the view is not necessarily bounded by f). We further assume that before the attack starts, the in-degrees and out-degrees of all correct nodes are (roughly) equal. This property is a close approximation of reality, since a benign gossip process preserves it [1].

Correct node u	Random correct node	Semantics
$x_u(t)/\tilde{x}_u(t)$	$x(t)/\tilde{x}(t)$	number/fraction of faulty ids in the view
$y_u(t)/\tilde{y}_u(t)$		number/fraction of instances among the views of correct nodes
$g_u^{\text{push}}(t)/\tilde{g}_u^{\text{push}}(t)$	$g^{\text{push}}(t)/\tilde{g}^{\text{push}}(t)$	number/fraction of correct ids pushed to the node
$r_u^{\text{push}}(t)/\tilde{r}_u^{\text{push}}(t)$	$r^{\text{push}}(t)/\tilde{r}^{\text{push}}(t)$	number/fraction of faulty ids pushed to the node
$g_u^{\text{pull}}(t)/\tilde{g}_u^{\text{pull}}(t)$	$g^{\text{pull}}(t)/\tilde{g}^{\text{pull}}(t)$	number/fraction of correct ids pulled by the node
$r_u^{\text{pull}}(t)/\tilde{r}_u^{\text{pull}}(t)$	$r^{\text{pull}}(t)/\tilde{r}^{\text{pull}}(t)$	number/fraction of faulty ids pulled by the node

Table 1: **Definition of common random variables.**

Adversarial behavior. The adversary's way to partition the overlay is through increasing its representation in the views of correct nodes. We assume the worst-case behavior by faulty nodes. In particular, they push faulty ids to correct nodes and always return faulty ids to pulls.

We first bound the damage that can be caused within a *single* round (a similar approach was taken, e.g., in [20]). In Appendix B, we prove Lemma B.1, which asserts that in any single round, a *balanced* attack, which spreads faulty pushes evenly among correct nodes, maximizes the expected system-wide fraction of faulty ids, $\tilde{x}(t)$, among all strategies. In Section 5.1, we prove that if this attack persists, the ratio of faulty ids in the system eventually stabilizes at a fixed point. We study the convergence process, and show that for certain parameter choices, this fixed point is strictly smaller than 1.

Alternatively, an adversary can try to partition the network (rather than increase its representation) by targeting a subset of nodes with more pushes than in a balanced attack. Without prior information about the overlay's topology, attacking a single node can be most damaging, since the sets of edges adjacent to single nodes are likely to be the sparsest cuts in the overlay. Section 5.2 shows that had Brahms not used history samples, correct nodes could have been isolated in this manner. However, Brahms sustains such *targeted* attacks, even if they start immediately upon a node's join, when it is not represented in other views and has no history. The key property is that Brahms's gossip prevents isolation long enough for history samples to become effective.

Simulation setup. We validate our assumptions using simulations with N=1000 nodes or more. Each data point is averaged over 100 runs. The maximal possible number of faulty nodes, fN, remain always active. For simplicity, $p = f$. A different subset of faulty nodes push their ids to a given correct node in each round, using a round-robin schedule. Faulty nodes always respond to probe requests, to avoid invalidation.

5.1 Balanced Attack

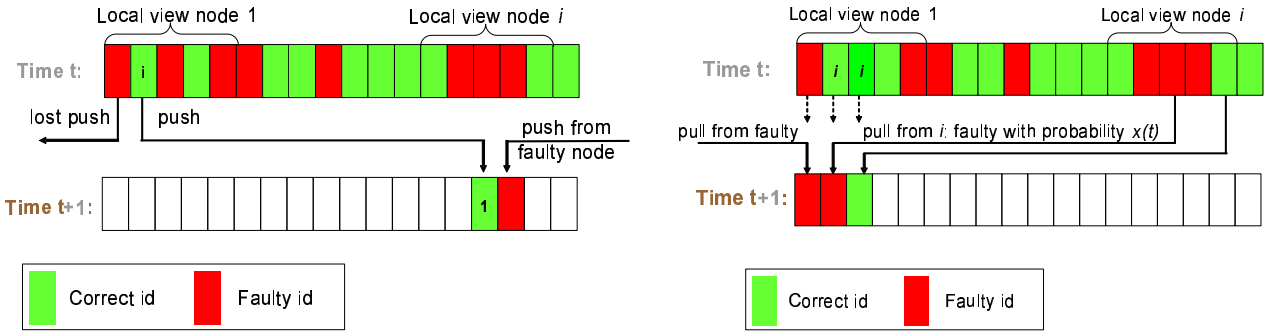
In the analysis of a balanced attack we ignore blocking since its only effect is to slow the convergence rate. Simulations show that this assumption has little effect on the results. Since a balanced attack does not distinguish between correct nodes, we assume that it preserves the in-degrees and out-degrees of all correct nodes equal over time:

Assumption 5.1 *For all $u \in \mathcal{C}(T_0)$ and all $t \geq T_0$: $x_u(t) = x(t)$, and $y_u(t) = \ell_1 - x_u(t)$.*

We show the existence of a parameter-dependent fixed point of $\tilde{x}(t)$ and the system's convergence to it. Since the focus is on asymptotic behavior, we assume $t \gg T_0$.

Lemma 5.1 *For $t \gg T_0$, if $p \neq 0$ or $\tilde{x}(t) \neq 1$, the expected system-wide fraction of faulty ids evolves as*

$$\mathrm{E}(\tilde{x}(t+1)) = \alpha \frac{p}{p + (1-p)(1-\tilde{x}(t))} + \beta(\tilde{x}(t) + (1-\tilde{x}(t))\tilde{x}(t)) + \gamma f.$$



(a) Impact of push (b) Impact of pull

Figure 5: **Fixed point analysis illustration.**

Proof : Consider the re-computation of $\mathcal{V}$ at a correct node u at time t. The weights of pushes, pulls, and history samples in the recomputed view are α, β and γ, respectively. Since the random selection process preserves the distribution of faulty ids in each data source, the probability of a push- (resp., pull)-originated entry being faulty is equal to the probability of receiving a faulty push (resp., pulling a faulty id).

Figure 5(a) illustrates the analysis of $\tilde{r}^{\text{push}}(t)$. Each correct node wastes an expected fraction $\tilde{x}(t)$ of its pushes because they are sent to faulty nodes. The rest are sent with an equal probability over each outgoing edge in $\mathcal{V}(t)$. Since out-degrees and in-degrees are equal among all correct nodes, each correct node u receives the same expected number of correct pushes: $\mathrm{E}(g_u^{\text{push}}(t)) = (1-\tilde{x}(t))\alpha\ell_1$. The variable $g_u^{\text{push}}(t)$ is binomially distributed, with the number of trials equal to the total number of pushes among all nodes with an outgoing edge to u. Since this number is large, the number of received correct pushes is approximately equal among all correct nodes, i.e., $g_u^{\text{push}}(t) \approx (1-\tilde{x}(t))\alpha\ell_1$, for all u.

The total number of correct pushes is $\alpha\ell_1|\mathcal{C}(T_0)|$, which is $1-p$ out of all pushes, hence the total number of faulty pushes is $\frac{p\alpha\ell_1}{1-p}|\mathcal{C}(T_0)|$. Therefore, u receives exactly $r_u^{\text{push}}(t) = \frac{p}{1-p}\alpha\ell_1$ faulty pushes, i.e., their fraction among all received pushes is:

$$\tilde{r}_u^{\text{push}}(t) = \frac{\frac{p}{1-p}\alpha\ell_1}{\frac{p}{1-p}\alpha\ell_1 + (1-\tilde{x}(t))\alpha\ell_1} = \frac{p}{p+(1-p)(1-\tilde{x}(t))}.$$

Hence, the expected ratio of push-originated faulty ids in $\mathcal{V}_u$ is $\alpha\frac{p}{p+(1-p)(1-\tilde{x}(t))}$.

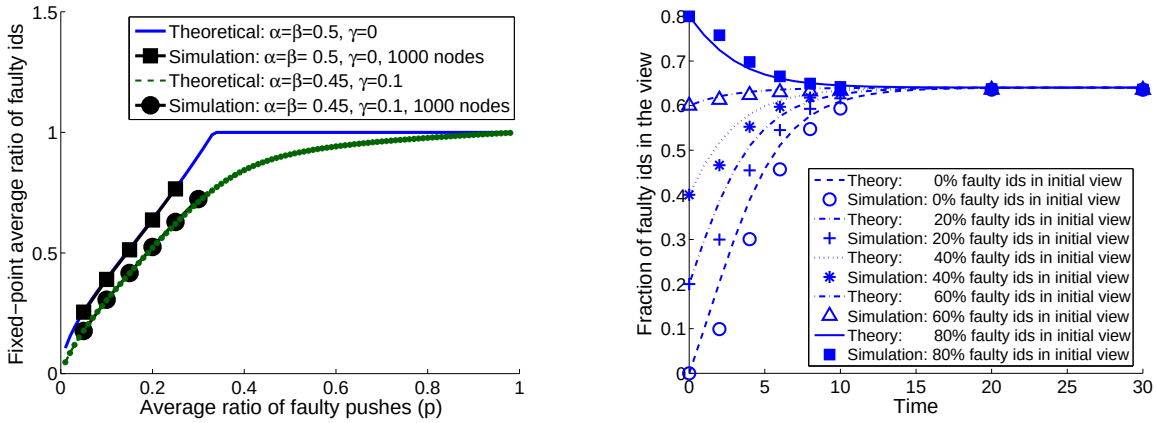
Figure 5(b) depicts the evolution of pull-originated faulty ids. Since all correct nodes have an equal out-degree, a correct node is pulled with probability $1-\tilde{x}(t)$, while a faulty node is pulled with probability $\tilde{x}(t)$. A pulled id is faulty with probability $\tilde{x}(t)$ if it comes from a correct node, and otherwise, it is always faulty. Hence, the expected fraction of pull-originated faulty ids is $\beta(\tilde{x}(t)+(1-\tilde{x}(t))\tilde{x}(t))$.

Finally, since $t \gg T_0$, the history sample is perfect (the ratio of faulty ids in it is f). Hence, its expected contribution is γf, and the claim follows. □

We now show that the system converges to a stable state. A value $\hat{x}$ is called a *fixed point* of $\tilde{x}(t)$ if $\mathrm{E}(\tilde{x}(t+1)) = \tilde{x}(t) = \hat{x}$. Substituting this requirement into the equation from Lemma 5.1, we get:

Lemma 5.2 *For $\alpha, \beta, \gamma, p, f \in [0,1]$, every real root $0 \le \hat{x} \le 1$ of the following cubic equation is a fixed point of $\tilde{x}(t)$, except for the root $x = 1$ for $p = 0$:*

$$\beta(1-p)\tilde{x}^3 + (2\beta p - 3\beta - p + 1)\tilde{x}^2 + (\gamma f p - \gamma f + 2\beta - 1)\tilde{x} + (\alpha p + \gamma f) = 0.$$



(a) Fixed point $\hat{x}$ as a function of p, for $\gamma = 0$ and $\gamma > 0$ (b) Convergence to $\hat{x}$: $N = 1000$, $p = 0.2$, $\alpha = \beta = 0.5$ and $\gamma = 0$.

Figure 6: **System-wide fraction of faulty ids in local views, under a balanced attack: (a) Fixed points (b) Convergence process.**

If $\gamma = 0$ (no history samples), $\hat{x} = 1$ is always a root. We call it a *trivial* fixed point. This is easily explainable, since if the views of all the correct nodes are totally poisoned, then neither pulls nor pushes help. Appendix B shows that if $\gamma = 0$, there can exist at most one nontrivial fixed point $0 \leq \hat{x} < 1$. For example, if $\alpha = \beta = \frac{1}{2}$ and $\gamma = 0$, then $\hat{x} = \frac{p+\sqrt{4p-3p^2}}{2(1-p)}$, for $0 \leq p \leq \frac{1}{3}$. In contrast, if the fraction of faulty pushes exceeds $\frac{1}{3}$, the only fixed point is 1, causing isolation of all correct nodes.

If $\gamma > 0$, there exists a single nontrivial fixed point for all p. This highlights the importance of history samples. Figure 6(a) depicts the analysis results, perfectly matched by simulations.

We conclude the analysis by proving convergence to a nontrivial fixed point.

Lemma 5.3 *If there exists a fixed point $\hat{x} < 1$ of $\tilde{x}(t)$, and $\tilde{x}(T_0) < 1$, then $\tilde{x}(t)$ converges to $\hat{x}$.*

Proof idea (see Appendix B). We show that for all t, the sequence of $\tilde{x}(t)$ is trapped between $\hat{x}$ and another sequence, $\phi(t)$, that converges to $\hat{x}$. Hillam's theorem [15] is then used to prove sequence convergence.

Since the balanced attack does not distinguish between correct nodes, the same result holds for $\tilde{x}_u(t)$, for each correct node u. Figure 6(b) depicts the convergence to the nontrivial fixed point from various initial values of $\tilde{x}(t)$. The analytical and simulation results are similar. The latter's convergence is slightly slower because the analysis ignores blocking.

5.2 Targeted Attack

We study a targeted attack on a single correct node u, which starts upon u's join at T_0. We prove that u is not isolated from the overlay by showing a lower bound on the expected time to isolation, which exceeds an upper bound on the time to a perfect correct sample (a sufficient condition for non-isolation, Section 4).

Lower bound on expected isolation time. As we seek a lower bound, we make a number of worst-case assumptions (formally stated in Appendix C). First, we analyze a simplified protocol that does not employ history samples (i.e., $\gamma = 0$), so that $\mathcal{S}$ does not correct $\mathcal{V}$'s bias. Next, we assume an unrealistic adaptive adversary that observes the exact number of correct pushes to u, $g_u^{\text{push}}(t)$, and complements them with $\alpha\ell_1 - g_u^{\text{push}}(t)$ faulty pushes – the most that can be accepted without blocking. The adversary maximizes its global representation through a balanced attack on all correct nodes $v \neq u$, thus minimizing the fraction

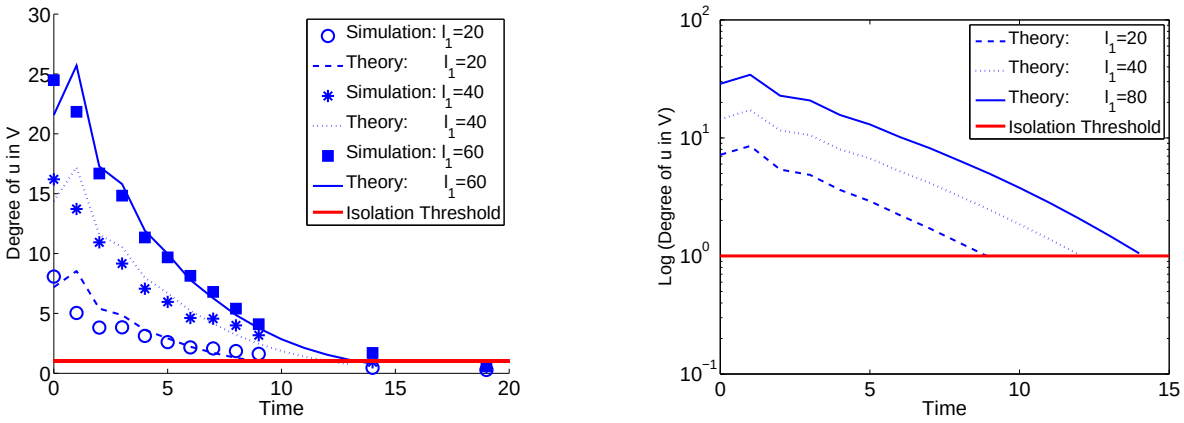
of correct ids that u pulls from correct nodes. Finally, we assume that u is not represented in the system initially, and it derives its initial view from a random set of correct nodes, where the ratio of faulty ids is at the fixed point (Section 5.1).

Clearly, the time to isolation in $\mathcal{V}(t)$ is a lower bound on that in $\mathcal{N}(t)$. We study the dynamics of the number of correct ids in u's out-degree in $\mathcal{V}(t)$, $\ell_1 - x_u(t)$, and u's in-degree, $y_u(t)$. We show that for any two specific values of $x_u(t)$ and $y_u(t)$, the expected out-degree and in-degree values at $t+1$ are

$$\begin{pmatrix} \ell_1 - \mathrm{E}(x_u(t+1)) \\ \mathrm{E}(y_u(t+1)) \end{pmatrix} = \begin{pmatrix} \beta(1-\hat{x}) & \alpha \\ \alpha\frac{1-p}{p+(1-p)(1-\hat{x})} & \beta(1-\hat{x}) \end{pmatrix} \times \begin{pmatrix} \ell_1 - x_u(t) \\ y_u(t) \end{pmatrix}.$$

Note that the coefficient matrix does not depend on $x_u(t)$ and $y_u(t)$, and the sum of entries in each row is smaller than 1. This implies that once the in-degree and the out-degree are close, they both decay exponentially. (Initially, this does not hold because u is not represented, i.e., $y_u(T_0) = 0$.) Therefore, the expected time to isolation is logarithmic with ℓ_1. Note that this process does not depend on the number of nodes, since blocking bounds the potential attacks on u independently of the system-wide budget of faulty pushes. Had blocking not been employed, the top right coefficient would have been 0 instead of α, because the adversary would have completely hijacked the push-originated entries in $\mathcal{V}_u$. The decay factor would have been much larger, leading to almost immediate isolation.

Figure 7(a) depicts the dynamics of u's expected degree (i.e., the sum of u's in- and out-degrees) until it becomes smaller than 1. Simulation results closely follow our analysis. The temporary growth in u's degree at $t = 1$ occurs because u becomes represented in the system after the first round. For example, the average time to isolation for $\ell_1 = 2\sqrt[3]{N}$ is 10 rounds. Figure 7(b) depicts the same results in log-scale, emphasizing the exponential decay of u's degree and the logarithmic dependency between ℓ_1 and time to isolation.

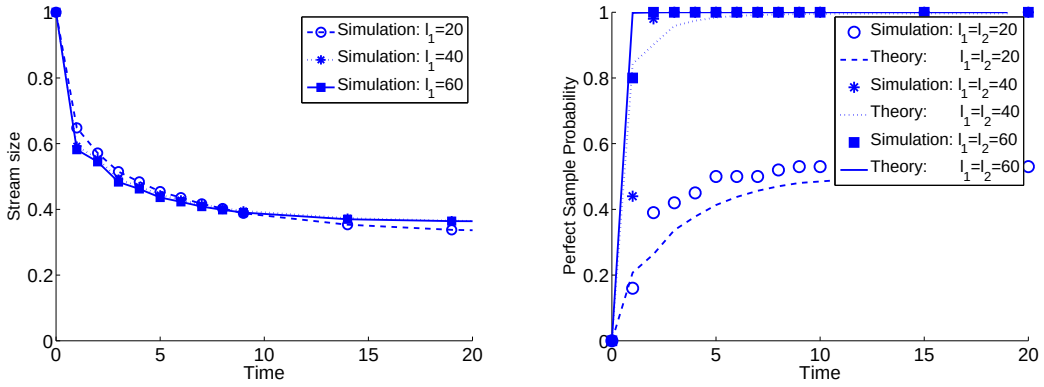


(a) Normal scale (theory and simulation), $N = 1000$

(b) Logarithmic scale (theory only), independent of N

Figure 7: **Targeted attack without history samples: node degree dynamics.** $N = 1000$**,** $p = 0.2$**,** $\alpha = \beta = 0.5$**,** $\gamma = 0$**. Without history samples a targeted attack isolates** u **in logarithmic time in** ℓ_1**.**

Upper bound on expected time to perfect correct sample. Lemma 4.3 bounds $PSP_u(t)$ for given values of the non-unique stream size $\Lambda(t)$, and the deficiency factor ρ (Section 4). The expected number of correct ids observed by u till the end of round T is $\Lambda(t) = \sum_{t=T_0}^{T_0+T-1}(\mathrm{E}(g_u^{\mathrm{push}}(t)) + \mathrm{E}(g_u^{\mathrm{pull}}(t)))$; the expected values of $g_u^{\mathrm{push}}(t)$ and $g_u^{\mathrm{pull}}(t)$ are by-products of the analysis in Appendix C, for $\gamma = 0$. Figure 8(a) depicts the deficiency factor ρ measured by our simulations, which behaves similarly for all values of ℓ_1: $\rho \geq 0.4$ for all t. Figure 8(b) depicts the progress of the upper bound of Lemma 4.3 with time, with $\Lambda(t)$ computed



(a) Deficiency factor (b) Perfect sample probability

Figure 8: **Dynamics within a targeted node ($N = 1000$, $p = 0.2$, $\alpha = \beta = 0.5$ and $\gamma = 0$): (a) Fraction of unique ids in the stream of correct ids, ρ. (b) Growth of Perfect Sample Probability (PSP) with time, $\rho = 0.4$. PSP becomes high quickly enough to prevent isolation.**

as explained above and $\rho = 0.4$. The corresponding simulation results show, for each time t, the fraction of runs in which at least one correct id in $\mathcal{S}_u$ is perfect. For $\ell_2 \geq 40$, the PSP becomes close to 1 in a few rounds, much faster than isolation happens (Figure 7(b)). For $\ell_1 = 20$, it stabilizes at 0.5. The growth stops because we run the protocol without history samples, thus u becomes isolated, and the id stream ceases. A higher PSP can be achieved by independently increasing ℓ_2, e.g., if ℓ_2 is 40, then the PSP grows to 0.8 (Figure 4.2). Note that perfect samples only provide an upper bound on self-healing time, as $\mathcal{S}_u$ contains imperfect correct ids, and u also becomes sampled by other correct nodes, w.h.p. These factors coupled with history samples ($\gamma > 0$) completely prevent u's isolation, as shown in Section 6.

6 Putting it All Together

In previous sections we analyzed each of Brahms's mechanisms separately. We now simulate the entire system. Figure 9 depicts the degree of node u in $\mathcal{N}(t)$ under a targeted attack. Node u remains connected to the overlay, thanks to history samples ($\gamma = 0.1$). Note that the actual degree of u in $\mathcal{N}(t)$ is higher than the lower bound shown in Section 5.2, due to the pessimistic assumptions made in the analysis (no history samples, no imperfect correct ids, etc.).

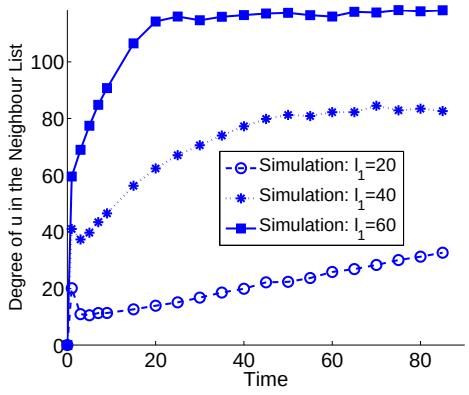
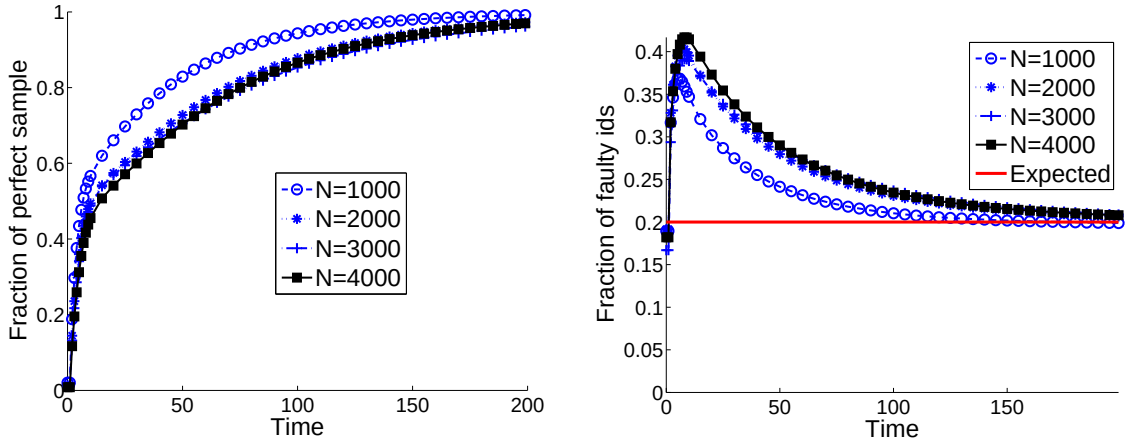


Figure 9: **Degree dynamics of an attacked node in $\mathcal{N}(t)$, $N = 1000$, $p = 0.2$, $\alpha = \beta = 0.45$ and $\gamma = 0.1$.**



(a) Perfect samples vs time

(b) Faulty ids vs. time

Figure 10: **Fraction of (a) perfect samples and (b) faulty nodes in $\mathcal{S}$, under a balanced attack ($f = 0.2$), for 1000, 2000, 3000 and 4000 nodes, $\ell_2 = 2\sqrt[3]{N}$.**

We now demonstrate the convergence of $\mathcal{S}$ in the correct nodes. We simulate systems with up to $N = 4000$ nodes; ℓ_1 and ℓ_2 are set to $2\sqrt[3]{N}$. To measure the quality of sample $\mathcal{S}$ under a balanced attack, we depict the fraction of ids in $\mathcal{S}$ that are indeed the perfect sample over time (Figure 10(a)). Note that this criterion is conservative, since missing a perfect sample does not automatically lead to a biased choice. More than 50% of perfect samples are achieved within less than 15 rounds; for $\ell_2 = \ell_1 = 3\sqrt[3]{N}$, the convergence is twice as fast. Figure 10(b) depicts the evolution of the fraction of faulty ids in $\mathcal{S}$. Initially, this fraction equals f, and at first increases, up to approximately the fixed point's value. This is to be expected, since the first observed samples are distributed like the original (biased) data stream. Subsequently, as the node encounters more unique ids, the quality of $\mathcal{S}$ improves, and the fraction of faulty ids drops fast to f. The protocol exhibits almost perfect scalability, as the convergence rate is the same for $N \geq 2000$.

7 Conclusions

We presented Brahms, a Byzantine-resilient membership sampling algorithm. Brahms stores small views, and yet resists the failure of a linear portion of the nodes. It ensures that every node's sample converges to a uniform one, which was not achieved before by gossip-based membership even in benign settings. We presented extensive analysis and simulations explaining the impact of various attacks on the membership, as well as the effectiveness of the different mechanisms Brahms employs.

Acknowledgments

We thank Christian Cachin for his valuable suggestions that helped to improve our paper. We are grateful to Roie Melamed and Igor Yanover for fruitful discussions of a random walk overlay-based solution.

References

[1] A. Allavena, A. Demers, and J. E. Hopcroft. Correctness of a gossip based membership protocol. In *ACM PODC*, pages 292–301, 2005.

[2] B. Awerbuch and C. Scheideler. Towards a Scalable and Robust DHT. In *SPAA*, pages 318–327, 2006.

[3] G. Badishi, I. Keidar, and A. Sasson. Exposing and Eliminating Vulnerabilities to Denial of Service Attacks in Secure Gossip-Based Multicast. In *DSN*, pages 201–210, June – July 2004.

[4] Z. Bar-Yossef, R. Friedman, and G. Kliot. RaWMS - Random Walk based Lightweight Membership Service for Wireless Ad Hoc Networks. In *ACM MobiHoc*, pages 238–249, 2006.

[5] K. P. Birman, M. Hayden, O. Ozkasap, Z. Xiao, M. Budiu, and Y. Minsky. Bimodal multicast. *ACM Transactions on Computer Systems*, 17(2):41–88, 1999.

[6] A. Z. Broder, M. Charikar, A. M. Frieze, and M. Mitzenmacher. Min-wise independent permutations. *J. Computer and System Sciences*, 60(3):630–659, 2000.

[7] T. Condie, V. Kacholia, S. Sankararaman, J. Hellerstein, and P. Maniatis. Induced Churn as Shelter from RoutingTable Poisoning. In *NDSS*, 2006.

[8] A. Demers, D. Greene, C. Hauser, W. Irish, J. Larson, S. Shenker, H. Sturgis, D. Swinehart, and D. Terry. Epidemic Algorithms for Replicated Database Management. In *ACM PODC*, pages 1–12, August 1987.

[9] D.Malkhi, Y. Mansour, and M. K. Reiter. On Diffusing Updates in a Byzantine Environment. In *Symposium on Reliable Distributed Systems*, pages 134–143, 1999.

[10] J.R. Douceur. The Sybil Attack. In *IPTPS*, 2002.

[11] P. Th. Eugster, R. Guerraoui, S. B. Handurukande, P. Kouznetsov, and A.-M. Kermarrec. Lightweight probabilistic broadcast. *ACM Transactions on Computer Systems (TOCS)*, 21(4):341–374, 2003.

[12] A. J. Ganesh, A.-M. Kermarrec, and L. Massoulie. SCAMP: Peer-to-Peer Lightweight Membership Service for Large-Scale Group Communication. In *Networked Group Communication*, pages 44–55, 2001.

[13] C. Gkantsidis, M. Mihail, and A. Saberi. Random walks in peer-to-peer networks. In *IEEE INFOCOM*, 2004.

[14] O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *JACM*, 33(4):792–807, 1986.

[15] B. P. Hillam. A Generalization of Krasnoselski's Theorem on the Real Line. *Math. Mag.*, 48:167–168, 1975.

[16] M. Jelasity, S. Voulgaris, R. Guerraoui, A.-M. Kermarrec, and M. van Steen. Gossip-based peer sampling. *ACM Trans. Comput. Syst.*, 25(3):8, 2007.

[17] H. Johansen, A. Allavena, and R. van Renesse. Fireflies: scalable support for intrusion-tolerant network overlays. In *Proc. of the 2006 EuroSys conference (EuroSys)*, pages 3–13, 2006.

[18] V. King and J. Saia. Choosing a random peer. In *ACM PODC*, pages 125–130, 2004.

[19] C. Law and K. Siu. Distributed construction of random expander networks. In *IEEE INFOCOM*, April 2003.

[20] H. C. Li, A. Clement, E. L. Wong, J. Napper, I. Roy, L. Alvisi, and M. Dahlin. BAR Gossip. In *Proc. of the 7th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pages 45–58, November 2006.

[21] C. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker. Search and replication in unstructured peer-to-peer networks. In *Proc. of the 16th International Conference on Supercomputing (ICS)*, pages 84–95, 2002.

[22] G. Manku, M. Bawa, and P. Raghavan. Symphony: Distributed hashing in a small world. In *Proc. of the 4th USENIX Symposium on Internet Technologies and Systems (USITS)*, 2003.

[23] L. Massoulie, E. Le Merrer, A.-M. Kermarrec, and A. J. Ganesh. Peer Counting and Sampling in Overlay Networks: Random Walk Methods. In *ACM PODC*, pages 123–132, 2006.

[24] R. Melamed and I. Keidar. Araneola: A Scalable Reliable Multicast System for Dynamic Environments. In *IEEE NCA*, pages 5–14, 2004.

[25] R. C. Merkle. Secure Communications over Insecure Channels. *CACM*, 21:294–299, April 1978.

[26] Y. M. Minsky and F. B. Schneider. Tolerating Malicious Gossip. *Dist. Computing*, 16(1):49–68, February 2003.

[27] Atul Singh, T.-W. Ngan, Peter Druschel, and Dan S. Wallach. Eclipse Attacks on Overlay Networks: Threats and Defenses. In *IEEE INFOCOM*, 2006.

[28] S. Voulgaris, D. Gavidia, and M. van Steen. CYCLON: Inexpensive Membership Management for Unstructured P2P Overlays. *Journal of Network and Systems Management*, 13(2):197–217, July 2005.

A Analysis - sampling

A.1 Eventual convergence

Proposition A.1 *If $\mathcal{N}(t)$ remains weakly connected for each $t \geq T_1$ for some $T_1 \geq T_0$ then, for each $u, v \in \mathcal{C}(T_0)$, there is a positive probability of $v \in \mathcal{V}_u(t'_i)$ for infinitely many times $t'_i > T_1$.*

Proof : For $t \geq T_1$, we define the (t)-*reachable set* of u, denoted $\Gamma_u(t)$, as a set of correct ids that have nonzero probability to appear in $\mathcal{V}_u(t`)$, for some $T_1 \leq t` \leq t$. Clearly, $\Gamma_u(T_1) = \mathcal{V}_u(T_1) \bigcap \mathcal{C}(T_0)$, and $\Gamma_u(t) \subseteq \Gamma_u(t+1)$, for all $t \geq T_1$. We show that as long as $\Gamma_u(t) \subset \mathcal{C}(T_0)$, the set $\Gamma_u(t)$ grows by at least one correct id each three rounds. For simplicity we consider a slightly transformed protocol that does not employ blocking. The only effect of this is a faster, by a constant factor, growth rate of Γ_u. Note that $v \in \Gamma_u(t)$ implies $\Pr(v \in \mathcal{V}_u(t')) > 0$ for all $t' \geq t$, as v can remain in $\mathcal{V}_u$ indefinitely, e.g., by repeatedly exchanging with u push messages, or if v is sampled by u into $\mathcal{S}_u$ and then returned by the history sampling mechanism to $\mathcal{V}_u$.

A new entry in $\mathcal{V}_u(t)$ can appear following (1) a push from some other node v, (2) pulling a view from some other node v, and (3) applying a history sample from $\mathcal{S}_u(t)$. Let us define the effects of these three operations as following: $\Gamma_u(t) = \Gamma_u(t-1) \bigcup \Delta(t) \bigcap \mathcal{C}(T_0)$, where

$$\Delta(t) = \Delta^{push}(t) \bigcup \Delta^{pull}(t) \bigcup \Delta^{history}(t)$$

is the set of nodes that can potentially reach $\mathcal{V}_u(t)$. We now describe each of its components.

$$\Delta^{push}(t) = \{v | u \in \mathcal{V}_v(t-1)\} \bigcup \{v | \Gamma_u(t-2) \cap \mathcal{V}_v(t-2) \neq \emptyset\} \bigcup \{v | \Gamma_u(t-3) \cap \mathcal{S}_v(t-3) \neq \emptyset\}$$

is a set of to all the node ids that can potentially reach u through push. Note that only the first term refers to the direct pushes to u. The second term refers to pushes to some intermediate node $w \in \Gamma_u(t-2)$, that can then be pulled by u. The second term refers to the nodes that first sample some intermediate node $w \in \Gamma_u(t-3)$ from their history sample, then push to w, and only then their ids can be pulled by u from w.

$$\Delta^{pull}(t) = \bigcup_{v \in \Gamma_u(t)} \mathcal{V}_v(t-1)$$

is a set of to all the nodes that u can potentially pull from.

$$\Delta^{history}(t) = \mathcal{S}_u(t-1)$$

is a set of to all the nodes that u can potentially sample from its history sample.

Recall that $\mathcal{N}(t)$ is a directed graph spanned by $(\mathcal{V}_v(t) \bigcup \mathcal{S}_v(t)) \bigcap \mathcal{C}(t)$ of all correct nodes v. Since $\mathcal{N}(t)$ is connected, for any subset of $\mathcal{C}(T_0)$, in particular $\Gamma_u(t)$, there exist at least one edge between that subset and the complementing subset. Consider a node $v \in \mathcal{C}(T_0) \setminus \Gamma_u(t)$, connected to some $w \in \Gamma_u(t)$ by an edge in $\mathcal{N}(t)$. It is easy to see that v will appear in either $\Delta(t+1)$, $\Delta(t+2)$, or $\Delta(t+3)$, depending on w and the origins of the edge (e.g., on whether $w = u$, whether the edge originated from $\mathcal{V}_u$ or $\mathcal{S}_U$, etc.). Therefore, for at least each third round t, $\Delta(t) \bigcap \Gamma_u(t) \neq \emptyset$, and $\Gamma_u(t)$ is a proper superset of $\Gamma_u(t-1)$, which guarantees that by $t = T_1 + 3|\mathcal{C}(T_0)|$, $\Gamma_u(t)$ will contain all the nodes in $\mathcal{C}(T_0)$.

We have shown that by the time $T_1 + 3|\mathcal{C}(T_0)|$, all ids in $\mathcal{C}(T_0)$ have a positive probability to appear in $\mathcal{V}_u$ between time T_1 and $T_1 + 3|\mathcal{C}(T_0)|$. Obviously, we can start over and see that after the next period of

$3|\mathcal{C}(T_0)|$ rounds all ids in $\mathcal{C}(T_0)$ had a chance to appear in $\mathcal{V}_u$, and so on. We conclude that after T_1, each id in $\mathcal{C}(T_0)$ can appear in $\mathcal{V}_u$ infinitely many times. □

The following proposition shows that each correct perfect id V_R can eventually be observed by R, so that there exist time t, such that $R(t) = V_R$.

Proposition A.2 *If $\mathcal{N}(t)$ remains weakly connected for each $t \geq T_1$ for some $T_1 \geq T_0$ then, for each $V_R \in \mathcal{C}(T_0)$*

$$\lim_{t\to\infty} \Pr(R(t) = V_R | V_R \in \mathcal{C}(T_0)) = 1.$$

Proof : Let node u be the owner of R. It follows from Proposition A.1 that the probability of V_R appear in $\mathcal{V}_u$, and consequently in u's stream approaches 1. The proposition follows immediately. □

W assume that each correct id has equal probability to take place of an invalidated faulty id in a Sampler.

Assumption A.1 *In each Sampler R, such that $V_R \notin \mathcal{C}(T_0)$, for each $v \in \mathcal{C}(T_0)$ and for each $t > T_0$,*

$$0 \leq \Pr(R(t) = v | V_R \notin \mathcal{C}(T_0)) \leq \frac{1}{|\mathcal{C}(T_0)|}.$$

The assumption is justified since the if faulty node V_R responds to all invalidation probes, $\Pr(R(t) = v) = 0$, and if it never responds to them, $\Pr(R(t) = v) = \frac{1}{|\mathcal{C}(T_0)|}$. Otherwise, if it sometimes does and sometimes does not, since faulty nodes do not adapt to the local choices of correct nodes, no correct id will be overrepresented compared to the other correct nodes.

Theorem 4.1 (restated) *If $\mathcal{N}(t)$ remains weakly connected for each $t \geq T_1$, for some $T_1 \geq T_0$, then, for all $v \in \mathcal{C}(T_0)$, and $\varepsilon > 0$, there exists $T_\varepsilon \geq T_1$ such that for all $t \geq T_\varepsilon$*

$$\frac{1-f}{|\mathcal{C}(T_0)|} - \varepsilon \leq \Pr(R(t) = v) \leq \frac{1}{|\mathcal{C}(T_0)|} + \varepsilon.$$

Proof : We can write $\Pr(R(t) = v)$ as following:

$$\begin{aligned} \Pr(R(t) = v) &= \Pr(R(t) = v | V_R \in \mathcal{C}(T_0)) \cdot \Pr(V_R \in \mathcal{C}(T_0)) \\ &+ \Pr(R(t) = v | V_R \notin \mathcal{C}(T_0)) \cdot \Pr(V_R \notin \mathcal{C}(T_0)). \end{aligned}$$

From Proposition A.2 we know that $\lim_{t\to\infty} \Pr(R(t) = V_R | V_R \in \mathcal{C}(T_0)) = 1$, so that

$$\lim_{t\to\infty} \Pr(R(t) = v | V_R \in \mathcal{C}(T_0)) = \Pr(V_R = v | V_R \in \mathcal{C}(T_0)) = \frac{1}{|\mathcal{C}(T_0)|}.$$

From here, for each $\varepsilon > 0$, there exists $T_\varepsilon \geq T_1$ such that for all $t \geq T_\varepsilon$,

$$\frac{1}{|\mathcal{C}(T_0)|} - \varepsilon \leq \Pr(R(t) = v | V_R \in \mathcal{C}(T_0)) \leq \frac{1}{|\mathcal{C}(T_0)|} + \varepsilon.$$

Using Assumption A.1, and since $\Pr(V_R \in \mathcal{C}(T_0)) \geq 1-f$ and $\Pr(V_R \in \mathcal{C}(T_0)) \leq f$, we bound $\Pr(R(t) = v)$ as following. For all $\varepsilon > 0$, there exists $T_\varepsilon \geq T_1$ such that for all $t \geq T_\varepsilon$

$$\frac{1-f}{|\mathcal{C}(T_0)|} - \varepsilon \leq Pr(R(t) = v) \leq \frac{1}{|\mathcal{C}(T_0)|} + \varepsilon.$$

□

A.2 Convergence rate

In the following lemma we study the dependency between the probability of a sampler to output a correct perfect id and the number $\Lambda(t)$ of correct ids observed by the Sampler, and the stream deficiency factor ρ.

Proposition A.3 *For $|\mathcal{C}(T_0)| \gg 1$ and for each $t > T_1$, for some $T_1 \geq T_0$,*

$$\Pr(R(t) = V_R | V_R \in \mathcal{C}(T_0)) = 1 - e^{-\frac{\rho\Lambda(t)}{|\mathcal{C}(T_0)|}}.$$

Proof : Sampler outputs its perfect id V_R only after that id passed in the Sampler's input stream. So the probability of $R(t) \neq V_R$ is the probability that V_R did not appear in the stream of during the rounds $T_0 \leq t' \leq t$. Denote element j (considering only the correct ids) in the input stream of R by $G(j)$, and note that for each $v \in \mathcal{C}(T_0)$, $\Pr(G(j) = v) = \frac{1}{|\mathcal{C}(T_0)|}$. Then,

$$\begin{aligned}
\Pr(R(t) \neq V_R | V_R \in \mathcal{C}(T_0)) &= \Pr(V_R \notin \bigcup_{j=1}^{\rho\Lambda(t)} G(j) | V_R \in \mathcal{C}(T_0)) = \\
&\prod_{j=1}^{\rho\Lambda(t)} \Pr(G(j) \neq V_R | V_R \in \mathcal{C}(T_0)) = \\
&\prod_{j=1}^{\rho\Lambda(t)} (1 - \Pr(G(j) = V_R | V_R \in \mathcal{C}(T_0))) = \\
&\prod_{j=1}^{\rho\Lambda(t)} \left(1 - \frac{1}{|\mathcal{C}(T_0)|}\right) = \\
&\left(1 - \frac{1}{|\mathcal{C}(T_0)|}\right)^{\rho\Lambda(t)}.
\end{aligned}$$

Since $\frac{1}{|\mathcal{C}(T_0)|} \ll 1$, we can rely on $1 - x \approx e^{-x}$ for $x \ll 1$ and approximate the above as following

$$\Pr(R(t) \neq V_R | V_R \in \mathcal{C}(T_0)) \approx \left(e^{-\frac{1}{|\mathcal{C}(T_0)|}}\right)^{\rho\Lambda(t)} = e^{-\frac{\rho\Lambda(t)}{|\mathcal{C}(T_0)|}}.$$

From now on, we assume $\frac{1}{|\mathcal{C}(T_0)|}$ is small enough, so we use equality. It is now obvious that

$$\Pr(R(t) = V_R | V_R \in \mathcal{C}(T_0)) = 1 - e^{-\frac{\rho\Lambda(t)}{|\mathcal{C}(T_0)|}}.$$

□

Lemma 4.3 (restated) *Let $u \in \mathcal{C}(T_0)$ be a random correct node. Then, for $t > T_0$,*

$$PSP_u(t) \geq 1 - \left((1-f)e^{-\frac{\rho\Lambda(t)}{|\mathcal{C}(T_0)|}} + f\right)^{\ell_2}.$$

Proof : Since ℓ_2 of u's Samplers are independent, the probability of each one to have a correct perfect id is

$$\Pr(V_R \in \mathcal{C}(T_0)) = \frac{|\mathcal{A}(T_0) \bigcap \mathcal{C}(T_0)|}{|\mathcal{A}(T_0)|} \geq 1 - f.$$

Similarly,

$$\Pr(V_R \notin \mathcal{C}(T_0)) = \frac{|\mathcal{A}(T_0) \setminus \mathcal{C}(T_0)|}{|\mathcal{A}(T_0)|} \leq f.$$

Based on Proposition A.3, the probability of $R(t)$ not being a correct perfect id is

$$\begin{aligned}\Pr(R(t) \neq V_R \vee V_R \notin \mathcal{C}(T_0)) &= \Pr(R(t) \neq V_R | V_R \in \mathcal{C}(T_0)) \Pr(V_R \in \mathcal{C}(T_0)) + \Pr(V_R \notin \mathcal{C}(T_0)) \\ &\leq (1-f)e^{-\frac{\rho\Lambda(t)}{|\mathcal{C}(T_0)|}} + f.\end{aligned}$$

We now calculate the perfect sample probability $PSP_u(t)$, which equals 1 minus the probability of each of ℓ_2 Samplers not outputting a correct perfect id.

$$\begin{aligned}PSP_u(t) &= 1 - \prod_{i=1}^{\ell_2} \Pr(R(t) \neq V_R \vee V_R \notin \mathcal{C}(T_0)) = \\ &1 - (\Pr(R(t) \neq V_R \vee V_R \notin \mathcal{C}(T_0)))^{\ell_2} \geq \\ &1 - \left((1-f)e^{-\frac{\rho\Lambda(t)}{|\mathcal{C}(T_0)|}} + f\right)^{\ell_2}.\end{aligned}$$

□

B Balanced Attack Analysis

B.1 Short-term Optimality

We now prove that in any single round, a balanced attack maximizes the expected system-wide fraction of faulty ids, $\tilde{x}(t)$, among all strategies. Consider a schedule $R : \mathcal{C}(T_0) \rightarrow \mathbb{N}$ that assigns a number of faulty pushes to each correct node at round t. A schedule is *balanced* if for every two correct nodes u and v, it holds that $|R(u) - R(v)| \leq 1$. Otherwise, the schedule is *unbalanced*. We prove that every unbalanced schedule is suboptimal. All balanced schedules are equally optimal, for symmetry considerations.

Lemma B.1 *If schedule R is unbalanced, then there exists another schedule that imposes a larger expected ratio of faulty ids than R in round $t+1$.*

Proof : Since a schedule of faulty pushes in round t does not affect the pulls in this round, it is enough to prove the claim for the push-originated ids. Consider two nodes, u and v, such that $R(u) > R(v) + 1$. Consider an alternative schedule R' that differs from R in moving a single push from u to v. Consider the change in the expected cumulative fraction of push-originated faulty ids in $\mathcal{V}_u(t+1)$ and $\mathcal{V}_v(t+1)$ after this shift (in the other nodes, the ratio of faulty ids does not change).

The probability of a push-originated view entry at node u being faulty, provided that $R(u)$ faulty pushes were received, is equal to the expected fraction of $R(u)$ among all pushes received by u. Note that $R(u)$ is set in advance, i.e., without knowing the number of received correct pushes, $g_u^{\text{push}}(t)$ (Section 2.1). Conditioning on the latter, we get:

$$\mathrm{E}(\tilde{r}_u^{\text{push}} | r_u^{\text{push}} = R(u)) = \sum_{G=1}^{|\mathcal{C}(T_0)|} \Pr[g_u^{\text{push}}(t) = G] \cdot \frac{R(u)}{R(u)+G}.$$

We need to show that

$$\mathrm{E}(\tilde{r}_u^{\mathrm{push}}|r_u^{\mathrm{push}} = R(u)-1)+\mathrm{E}(\tilde{r}_v^{\mathrm{push}}|r_v^{\mathrm{push}} = R(v)+1) > \mathrm{E}(\tilde{r}_u^{\mathrm{push}}|r_u^{\mathrm{push}} = R(u))+\mathrm{E}(\tilde{r}_v^{\mathrm{push}}|r_v^{\mathrm{push}} = R(v)),$$

i.e.,

$$\sum_{G=1}^{|\mathcal{C}(T_0)|} \Pr[g_u^{\mathrm{push}}(t) = G] \cdot \frac{R(u)-1}{R(u)-1+G} + \sum_{G=1}^{|\mathcal{C}(T_0)|} \Pr[g_v^{\mathrm{push}}(t) = G] \cdot \frac{R(v)+1}{R(v)+1+G}$$
$$\geq \sum_{G=1}^{|\mathcal{C}(T_0)|} \Pr[g_u^{\mathrm{push}}(t) = G] \cdot \frac{R(u)}{R(u)+G} + \sum_{G=1}^{|\mathcal{C}(T_0)|} \Pr[g_v^{\mathrm{push}}(t) = G] \cdot \frac{R(v)}{R(v)+G}.$$

Since all correct nodes have the same in-degree in $\mathcal{V}(t)$ (Assumption 5.1), $g_u^{\mathrm{push}}(t)$ and $g_v^{\mathrm{push}}(t)$ have identical (binomial) distributions. Hence, it is enough to show that

$$\frac{R(u)-1}{R(u)-1+G} + \frac{R(v)+1}{R(v)+1+G} \geq \frac{R(u)}{R(u)+G} + \frac{R(v)}{R(v)+G},$$

for all $G \geq 0$ and all $R(u) > R(v)+1 > 0$. We start simplifying:

$$\frac{-G}{(R(u)-G)(R(u)-1+G)} + \frac{G}{(R(v)+G)(R(v)+1+G)} \geq 0.$$

Since $R(u)-1 \geq R(v)+1 > 0$,

$$\frac{-G}{(R(u)+G)(R(u)-1+G)} + \frac{G}{(R(v)+G)(R(v)+1+G)}$$
$$\geq \frac{-G}{(R(u)+G)(R(u)-1+G)} + \frac{G}{(R(u)-2+G)(R(u)-1+G)}$$
$$\geq \frac{G}{R(u)-1+G} \cdot [\frac{1}{R(u)-2+G} - \frac{1}{R(u)+G}] = \frac{G}{R(u)+G} \cdot \frac{2}{(R(u)+G)(R(u)-2+G)} > 0.$$

□

We conclude by showing that all balanced schedules are equally optimal for the adversary.

Proposition B.2 *Every two balanced schedules induce the same expected fraction of faulty ids in round* $t+1$.

Proof : R can be transformed into R' by a sequence of moves of a single push message from node u to node v, such that $R(u) = R(v)+1$ whereas $R'(v) = R'(u)+1$. For symmetry reasons, neither of these moves alters the expected cumulative fraction of faulty ids received by u and v. Hence, each transformation produces a schedule that implies the same $\tilde{x}(t+1)$ as the previous one. □

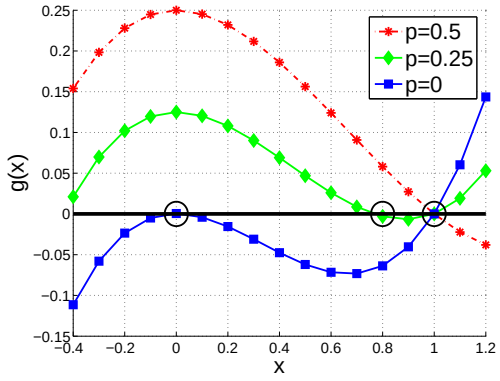


Figure 11: **Nontrivial fixed points $\hat{x}$ (depicted by circles), for $\alpha = \beta = \frac{1}{2}, \gamma = 0$.**

B.2 Fixed Point Analysis

Fixed point values. Consider $g(\tilde{x}) \triangleq \beta(1-p)\tilde{x}^3 + (2\beta p - 3\beta - p + 1)\tilde{x}^2 + (\gamma f p - \gamma f + 2\beta - 1)\tilde{x} + \alpha p + \gamma f$. By Lemma 5.2, the fixed point $\hat{x}$ is a root of $g(\tilde{x})$. Note that $g(0) = (\alpha + \gamma)p > 0$, and $g(1) = p(\alpha + \beta + \gamma p - 1) \leq 0$. Hence, if $\gamma > 0$, then the function has a single feasible root $\hat{x} \in (0, 1)$ (the others lie outside $[0, 1]$). In other words, there is always a single nontrivial fixed point. If $\gamma = 0$, then $\hat{x} = 1$ is always a root (a trivial fixed point). Since there exists an infeasible negative root, this leaves room for at most one more root $0 \leq \hat{x} < 1$ (i.e., there *may* exist at most one nontrivial fixed point). Figure 11 depicts the behavior of $g(x)$ for $\alpha = \beta = \frac{1}{2}$ ($\gamma = 0$), and different values of p. The fixed points are depicted by circles.

Two more parameter combinations deserve special interest:

1. $\beta = 1, \alpha = \gamma = 0$ (pull only, no history samples). The only valid root is $\hat{x} = 0$, for all p. That is, if none of the views initially contain a faulty id, and the faulty nodes cannot push their own ids, then the latter will remain unrepresented.

2. $\alpha = 1, \beta = \gamma = 0$ (push only, no history samples). The only valid root is $\hat{x} = \frac{p}{1-p}$, for $p \leq \frac{1}{2}$. That is, a nonzero fraction of correct ids can be maintained iff the majority of pushes are correct. This follows from the fact that a single correct push and a single faulty push equally contribute to the view.

Convergence. We prove convergence to a nontrivial fixed point.

Lemma 5.3 (restated) *If there exists a fixed point $\hat{x} < 1$ of $\tilde{x}(t)$, and $\tilde{x}(T_0) < 1$, then $\tilde{x}(t)$ converges to $\hat{x}$.*

Proof : We define $\psi(\tilde{x}) : [0, 1] \to [0, 1]$ as $\psi(\tilde{x}) \triangleq \alpha \frac{p}{p+(1-p)(1-\tilde{x})} + \beta(\tilde{x} + (1-\tilde{x})\tilde{x}) + \gamma f$. The sequence of expected values of $\tilde{x}(t)$ is defined by the iteration scheme $\{\tilde{x}(t+1) = \psi(\tilde{x}(t))\}$, for $t \geq T_0$. We show that for any $\tilde{x}(T_0) < 1$, this scheme converges to $\hat{x}$. For this purpose, we define an auxiliary sequence $\{\phi(t)\}$ that converges to $\hat{x}$, such that for each t, the value of $\tilde{x}(t)$ is trapped between $\hat{x}$ and $\phi(t)$, thus implying the desired result.

A straightforward calculus shows two facts to be used throughout the proof:

1. $\psi(\tilde{x})$ is monotonically increasing for $\tilde{x} \in [0, 1]$, since both $\frac{p}{p+(1-p)(1-\tilde{x})}$ and $\tilde{x} + (1-\tilde{x})\tilde{x} = 2\tilde{x} - \tilde{x}^2$ are monotonically increasing in this interval.

2. The absolute value of the first derivative of $\psi(\tilde{x})$ for $x \in [0,1]$ is bounded by a constant K (except for a combination $p = 0, x = 1$ which we do not consider).

By the mean value theorem, for all $\tilde{x}_1, \tilde{x}_2 \in [0,1]$ ($\tilde{x}_1 \leq \tilde{x}_2$), there exists $\tilde{x}' \in [\tilde{x}_1, \tilde{x}_2]$ such that

$$\psi(\tilde{x}_2) - \psi(\tilde{x}_1) = \frac{\delta\psi}{\delta x}(x') \cdot (\tilde{x}_2 - \tilde{x}_1).$$

Hence, the function satisfies the Liphschitz condition with constant K, namely, for each pair $\tilde{x}_1, \tilde{x}_2 \in [0,1]$, it holds that $|\psi(\tilde{x}_1) - \psi(\tilde{x}_2)| \leq K|\tilde{x}_1 - \tilde{x}_2|$. Therefore, by Hillam's theorem [15], the iteration scheme $\{\phi(t+1) = \lambda\phi(t) + (1-\lambda)\psi(\phi(t))\}$, where $\lambda = \frac{1}{K+1}$, converges to a fixed point of $\psi(x)$ for each $\phi(T_0) \in [0,1]$. It remains to show that the sequence $\{\tilde{x}(t)\}$ is confined between $\hat{x}$ and $\{\phi(t)\}$, and therefore, it also converges to $\hat{x}$. Specifically, we argue that:

1. Assume that $\hat{x} < \tilde{x}(T_0) = \phi(T_0) < 1$. Then, (a) the sequence $\{\phi(t)\}$ converges to $\hat{x}$, and (b) for all $t \geq T_0$, it holds that $\hat{x} < \tilde{x}(t) \leq \phi(t)$.

2. Assume that $0 \leq \tilde{x}(T_0) = \phi(T_0) < \hat{x}$. Then, (a) the sequence $\{\phi(t)\}$ converges to $\hat{x}$, and (b) for all $t \geq T_0$, it holds that $\phi(t) \leq \tilde{x}(t) < \hat{x}$.

We prove the first part of the claim (the second part's proof is symmetrical). Recall that $\hat{x}$ is a single nontrivial fixed point. By the definition, $\hat{x}$ is the root of the function $\psi(x) - x$, which is negative for $x \in (\hat{x}, 1)$ (i.e., $\psi(x) < x$). For an arbitrary $x \in (\hat{x}, 1)$, it holds that $\lambda x + (1-\lambda)\psi(x) < x$, i.e, the sequence $\{\phi(t)\}$ is monotonically decreasing with t. Hence, this sequence cannot converge to the trivial fixed point (if one exists), i.e., it converges to $\hat{x}$.

Next, we prove that $\hat{x} < \tilde{x}(t) \leq \phi(t)$ by induction on t. The basis is immediate. Assume that $\hat{x} < \tilde{x}(t) \leq \phi(t)$ for some $t \geq T_0$. We denote $X \triangleq \tilde{x}(t)$ and $\Phi \triangleq \phi(t)$. It holds that $\phi(t+1) = \lambda\Phi + (1-\lambda)\psi(\Phi) > \psi(\Phi)$. Since $\psi(x)$ is a monotonically increasing function for $x \in [0,1]$, $\psi(\Phi) \geq \psi(X) = \tilde{x}(t+1)$, that is, $\phi(t+1) > \tilde{x}(t+1)$. Similarly, $\tilde{x}(t+1) = \psi(X) > \psi(\hat{x}) = \hat{x}$, thus concluding the induction step. □

C Targeted Attack Analysis

This section analyzes the dynamics of a targeted attack on a single correct node, which aims isolating it from the other correct nodes.

C.1 Assumptions

We use the following assumptions on the environment to bound the time to isolation from below.

Assumption C.1 *(no history samples) $\gamma = 0$, which is equivalent to the worst-case assumption that the expected ratio of faulty ids in $\mathcal{S}$ at all times is equal to that in the id stream observed by the node (i.e., history samples are ineffective).*

Assumption C.2 *(unrealistically strong adversary) In each round $t \geq T_0$, the adversary observes the exact number of correct pushes received by u, $g_u^{\text{push}}(t)$, and complements it with faulty pushes to $\alpha\ell_1$ (i.e., the maximal number of faulty ids that can be accepted without blocking). Formally, $r_u^{\text{push}}(t) \triangleq \max(\alpha\ell_1 - g_u^{\text{push}}(t), 0)$.*

Assumption C.3 *(background attack on the rest of the system) The adversary maximizes its global representation through a balanced attack on all correct nodes $v \neq u$. At time T_0, the system-wide expected fraction of faulty ids is at the fixed point $\hat{x}$. (Note that this attack minimizes the fraction of correct ids that u can pull from correct nodes).*

Assumption C.4 *(fresh attacked node) u joins the system at T_0. It is initially not represented in any correct node's view and u's initial view is taken from a random correct node.*

We assume that the effect of u on the entire system's dynamics is negligible. Hence, we assume that the out-degrees and the in-degrees of all correct nodes except u are equal at all times (Assumption 5.1), and these nodes do not block (Section 5.1 showed that the system-wide effect of blocking is marginal).

C.2 Node Degree Dynamics

We study the dynamics of the degree of the attacked node u $\mathcal{V}(t)$. Consider a set of triples $\{(X, Y, t)\}$, each standing for a state $\{x_u(t) = X \ \wedge \ y_u(t) = Y\}$, for $X \in \{0, \ldots, \ell_1\}, Y \in \{0, \ldots, |\mathcal{C}(T_0)|\ell_1\}$. Each t defines a probability space, i.e., $\sum_{X,Y} \Pr[(X, Y, t)] = 1$. Since u is initially not represented, the only states that have non-zero probability for $t = T_0$ are those for which $Y = 0$. The probability distribution over these states is identical to the distribution of $x_u(T_0)$. Since u borrows its initial view from a random collection of correct nodes, $x_u(T_0) \sim \text{Bin}(\ell_1, \hat{x})$.

We now develop probability spaces for each $t > T_0$. The notation $\Pr[(X', Y', t+1)|(X, Y, t)]$ stands for the probability of transition from state (X, Y, t) to state (X', Y', t). That is, $\Pr[(X', Y', t+1)] = \sum_{X,Y} \Pr[(X', Y', t+1)|(X, Y, t)] \cdot \Pr[(X, Y, t)]$. To analyze $\Pr[(X', Y', t+1)|(X, Y, t)]$ we separately consider four independent random variables: the number of push- and pull-originated entries in $\mathcal{V}_u$, (denoted $x_u^{\texttt{push}}(t)$ and $x_u^{\texttt{pull}}(t)$), and the number of push- and pull-propagated instances of u in the views of correct nodes (denoted $y_u^{\texttt{push}}(t)$ and $y_u^{\texttt{pull}}(t)$). The first two affect X' whereas the last two affect Y'. We now demonstrate how conditional probability distributions for these variables are computed. For convenience, we omit the conditioning on (X, Y, t) from further notation.

$y_u^{\texttt{pull}}(t)$: Since the system is at the fixed point, the probability of pulling from some other correct node is $(1 - \hat{x})$. Hence, $y_u^{\texttt{pull}}(t+1)$ is a binomially distributed variable, with the number of trials equal to the total number of correct pulls, $(1 - \hat{x})\beta\ell_1|\mathcal{C}(T_0)|$, and the probability of success equal to the chance of an entry in a random node's view being u, namely $\frac{Y}{\ell_1|\mathcal{C}(T_0)|}$: $y_u^{\texttt{pull}}(t+1) \sim \text{Bin}((1 - \hat{x})\beta\ell_1|\mathcal{C}(T_0)|, \frac{Y}{\ell_1|\mathcal{C}(T_0)|})$. Note that $\text{E}(y_u^{\texttt{pull}}(t+1)) = \beta(1 - \hat{x})Y$.

$y_u^{\texttt{push}}(t)$: By Lemma 5.1, the number of pushes that reach correct nodes is $\alpha\ell_1|\mathcal{C}(T_0)|\frac{(1-\hat{x})(1-p)+p}{1-p}$. Denote the number of pushes from u to correct nodes in round t by $z_u(t)$. This is a binomially distributed variable with $\alpha\ell_1$ trials and probability of success equal to $1 - \frac{X}{\ell_1}$: $z_u(t) \sim \text{Bin}(\alpha\ell_1, 1 - \frac{X}{\ell_1})$. For a given $z_u(t) = Z$, since the total number of push-originated entries is $\alpha\ell_1|\mathcal{C}(T_0)|$, the number of push-propagated instances of u is $y_u^{\texttt{push}}(t+1|Z) \sim \text{Bin}\ (\alpha\ell_1|\mathcal{C}(T_0)|, \frac{Z}{\alpha\ell_1|\mathcal{C}(T_0)|((1-\hat{x})+\frac{p}{1-p})})$. Note that $\text{E}(y_u^{\texttt{push}}(t+1|Z)) = Z\frac{1-p}{p+(1-p)(1-\hat{x})}$. Hence, since Z is independent on p and $\hat{x}$,

$$\text{E}(y_u^{\texttt{push}}(t+1)) = \text{E}(Z)\frac{1-p}{p+(1-p)(1-\hat{x})} = \alpha(\ell_1 - X) \cdot \frac{1-p}{p+(1-p)(1-\hat{x})}.$$

$x_u^{\texttt{pull}}(t)$: A pull from a faulty node (which happens with probability $\frac{X}{\ell_1}$) produces a faulty id with probability 1, otherwise the probability to receive a faulty id is $\hat{x}$. Hence, the probability of pulling a faulty

id is $\frac{X}{\ell_1} + (1 - \frac{X}{\ell_1})\hat{x}$. That is, the number of pull-originated faulty ids in u's view is $x_u^{\texttt{pull}}(t+1) \sim \text{Bin}(\beta\ell_1, \frac{X}{\ell_1} + (1 - \frac{X}{\ell_1})\hat{x})$ (i.e., $\text{E}(x_u^{\texttt{pull}}(t+1)) = \beta(X + (\ell_1 - X)\hat{x})$).

We also compute the expected number of correct ids (with duplicates) pulled by u, which we need for estimating the size of the id stream that traverses this node (Section 5.2). Since u performs $\beta\ell_1$ pulls, and the expected number of correct ids pulled from a random node is $(1-\hat{x})\ell_1$,

$$\text{E}(g_u^{\text{pull}}(t)) = (1 - \frac{X}{\ell_1}) \cdot \beta\ell_1 \cdot (1-\hat{x})\ell_1 = (1-\hat{x})\ell_1(\ell_1 - X).$$

$x_u^{\texttt{push}}(t)$: The number of push-originated ids, $x_u^{\texttt{push}}(t+1)$, depends on the number of correct pushes received by u, $g_u^{\text{push}}(t)$. The latter is a binomially distributed variable, with the number of trials equal to the total number of correct pushes, $\alpha\ell_1|\mathcal{C}(T_0)|$, and the probability of success equal to the chance of an entry in a random node's view being u, namely $\frac{Y}{\ell_1|\mathcal{C}(T_0)|}$: $g_u^{\text{push}}(t) \sim \text{Bin}(\alpha\ell_1|\mathcal{C}(T_0)|, \frac{Y}{\ell_1|\mathcal{C}(T_0)|})$ (Note that $\text{E}(g_u^{\text{push}}(t)) = \alpha Y$. This value is of independent use for evaluating the size of the id stream that traverses u (Section 5.2)).

An expected representation of a correct node different from u in the system is $(1-\hat{x})\ell_1$. Since u is under-represented ($Y < (1-\hat{x})\ell_1$ w.h.p), the probability of receiving above $\alpha\ell_1$ correct pushes is low, and hence, we ignore the case of u being blocked by exceedingly many correct pushes. On the other hand, faulty pushes cannot block u either (Assumption C.2), and therefore, we assume that u never blocks. If $G \leq \alpha\ell_1$ correct pushes are received, the adversary complements the number of pushes to the maximum allowed (Assumption C.2), i.e., the fraction of faulty pushes to u is $1 - \frac{G}{\alpha\ell_1}$. Hence, the number of push-originated faulty ids in u's view is $x_u^{\texttt{push}}(t+1|G) \sim \text{Bin}(\alpha\ell_1, 1 - \frac{G}{\alpha\ell_1})$. In other words,

$$\text{E}(x_u^{\texttt{push}}(t+1)) = \alpha\ell_1(1 - \frac{\text{E}(g_u^{\text{push}}(t))}{\alpha\ell_1}) = \alpha\ell_1(1 - \frac{\alpha Y}{\alpha\ell_1}) = \alpha(\ell_1 - Y).$$

Putting it all together. Summing up, the expected values of in-degree and out-degree can be written as

$$\begin{pmatrix} \ell_1 - \text{E}(x_u(t+1)) \\ \text{E}(y_u(t+1)) \end{pmatrix} = \begin{pmatrix} \ell_1 - (\text{E}(x_u^{\texttt{push}}(t+1)) + \text{E}(x_u^{\texttt{pull}}(t+1))) \\ \text{E}(y_u^{\texttt{push}}(t+1)) + \text{E}(y_u^{\texttt{pull}}(t+1)) \end{pmatrix} =$$
$$= \begin{pmatrix} \ell_1 - (\alpha(\ell_1 - Y) + \beta(X + (\ell_1 - X)\hat{x})) \\ \alpha(\ell_1 - X)\frac{1-p}{p+(1-p)(1-\hat{x})} + \beta(1-\hat{x}) \end{pmatrix} =$$
$$= \begin{pmatrix} \beta(1-\hat{x}) & \alpha \\ \alpha\frac{1-p}{p+(1-p)(1-\hat{x})} & \beta(1-\hat{x}) \end{pmatrix} \cdot \begin{pmatrix} \ell_1 - x_u(t) \\ y_u(t) \end{pmatrix}$$

Since we have shown that u does not block w.h.p., and Section 5.1 demonstrated that the effect of blocking on the rest of correct nodes is negligible, we assume that all views are recomputed in each round. That is,

$$\Pr[x_u(t+1) = X' | (X,Y,t)] = \sum_{X_1' + X_2' = X'} \Pr[x_u^{\texttt{push}}(t) = X_1' | (X,Y,t)] \cdot \Pr[x_u^{\texttt{pull}}(t) = X_2' | (X,Y,t)],$$

and

$$\Pr[y_u(t+1) = Y' | (X,Y,t)] = \sum_{Y_1' + Y_2' = Y'} \Pr[y_u^{\texttt{push}}(t) = Y_1' | (X,Y,t)] \cdot \Pr[y_u^{\texttt{pull}}(t) = Y_2' | (X,Y,t)].$$

Since the computations of X' and Y' are independent, we conclude:

$$\Pr[(X',Y',t) | (X,Y,t)] = \Pr[x_u(t+1) = X' | (X,Y,t)] \cdot \Pr[y_u(t+1) = Y' | (X,Y,t)].$$