Δ-Reliable Broadcast: A Probabilistic Measure of Broadcast Reliability[*]

Patrick Th. Eugster[†]
Sun Microsystems
CH-8604 Volketswil, Switzerland

Rachid Guerraoui Petr Kouznetsov
Distributed Programming Laboratory, EPFL
CH-1015 Lausanne, Switzerland

Abstract

This paper introduces a new probabilistic specification of reliable broadcast communication primitives, called Δ-Reliable Broadcast. *This specification captures in a precise way the reliability of practical broadcast algorithms that, on the one hand, were devised with some form of reliability in mind but, on the other hand, are not considered reliable according to "traditional" reliability specifications.*

We illustrate the use of our specification by precisely measuring and comparing the reliability of two popular broadcast algorithms, namely Bimodal Multicast *and* IP Multicast. *In particular, we quantify how the reliability of each algorithm scales with the size of the system.*

1. Introduction

The growing interest in peer-to-peer computing has underlined the need for reliable broadcast algorithms deployable at large scale. Traditionally, the reliability of broadcast algorithms has been defined by three properties [6]:

Integrity For any message m, every correct process delivers m at most once, and only if m was previously broadcast by sender(m).

Validity If a correct process p broadcasts a message m, then p eventually delivers m.

Agreement If a correct process delivers a message m, then every correct process eventually delivers m.

To obtain these strong properties in a system with process and link failures, one employs costly, traditionally acknowledgement-based algorithms. These can be effective in a local environment, but may give unstable or unpredictable performance under stress, and hence tolerate limited scalability.

More pragmatic approaches to broadcast focus on performance in very large-scale settings, and sacrifice strong reliability guarantees (in the sense of [6]) to performance. Examples include the Internet *Multicast Usenet* (MUSE) protocol [7], or a broad range of so-called *network-level* protocols building on *IP Multicast* [3] (e.g., *Reliable Multicast Transport* [8]). The reliability of such protocols is typically expressed in *best-effort* terminology: if a participant discovers a failure, the "most reasonable" effort is made to overcome it, but there is no guarantee that such an attempt will be successful. In short, best-effort reliable algorithms are simply not intended to satisfy the traditional properties of Reliable Broadcast [6].

Birman et al [2] proposed a new look at broadcast reliability. In the context of their *gossip-based Bimodal Multicast* algorithm, they characterized a *useful* reliable broadcast algorithm through a set of properties including the following:

Atomicity *The protocol provides a bimodal delivery guarantee, under which there is a high probability that each broadcast will reach almost all processes, a low probability that each broadcast will reach just a very small set of processes, and a vanishingly small probability that it will reach some intermediate number of processes. That is, the traditional atomic "all or nothing" guarantee becomes "almost all or almost none".*

This property is very appealing from a practical viewpoint, but still rather informal.

The aim of this work is to introduce a precise *measure* to quantify the intuitively understandable notion of reliability used in practice. In other terms, we do not aim at introducing an original broadcast algorithm which would be more reliable than others, but at defining what the very statement "more reliable" may mean.

To this end, we introduce a new *probabilistically flavored*, *non-binary*, specification of the reliability of broadcast algorithms called Δ-*Reliable Broadcast*. Through this specification, we contribute to bridging the gap between theory and practice in broadcast reliability.

In short, Δ-*Reliability* measures a *probability distribution* for the *reliability degree* of a broadcast algorithm. The

[*] This work is partially supported by the Swiss National Science Foundation under grant 2100-064994.01/1, and by the EU IST OFES No. 01.0227 "PEPITO" project.

[†] Former affiliation: Distributed Programming Laboratory, EPFL, patrick.eugster@epfl.ch

use of probabilities enables the capture, to a certain extent, of the nondeterminism inherent to large-scale systems.

We illustrate the use our measure through two well-known examples. The first one, Bimodal Multicast [2], is a representative of the rapidly proliferating family of gossip-based algorithms which have received much attention lately, precisely because they are "pretty reliable". As a representative of the class of best-effort algorithms often used in practice, namely the network-level protocols, we discuss IP Multicast [3] on top of which many other "reliable" network-level broadcast protocols are built.

We also demonstrate the use of Δ-Reliability in comparing broadcast algorithms by contrasting Bimodal Multicast and IP Multicast, confirming that, in most practical environments, Bimodal Multicast is "more reliable" than IP Multicast, especially as the system grows in size. This is insofar unsurprising as IP Multicast has not been designed to be reliable, yet illustrates the usefulness of our specification in.quantifying the difference between algorithms.

The practical use of our Δ-Reliability measure is furthermore illustrated through the scalability analysis of Bimodal Multicast which illuminates very attractive scalability properties of the algorithm.

Roadmap. Section 2 introduces Δ-Reliability. Section 3 discusses the Δ-Reliability of Bimodal Multicast. Section 4 similarly applies our specification of Δ-Reliability to IP Multicast. Section 5 illustrates the use of Δ-Reliability in comparing broadcasting algorithms through Bimodal Multicast and IP Multicast. Section 6 concludes with final remarks, also on the applicability of our specification.

2. Δ-Reliable Broadcast: specification

This section presents our approach to measuring, in a probabilistic sense, the reliability of a broadcast algorithm. (Alternatives are discussed in [4].)

2.1. System and environment

We consider an asynchronous (in the sense of [6]) system Π of processes $\{p_1, .., p_n\}$. Processes are connected through fair lossy channels of infinite capacity. Let m be any message, uniquely identified and equipped, in particular, with a parameter $sender(m)$. Processes communicate by message passing defined by the primitives $send(m)$ and $receive(m)$. Broadcast is defined by the primitives $broadcast(m)$ and $deliver(m)$. Processes are subject to *crash* failures. A *correct* (in a given algorithm run) process is one that never crashes (in that run). To simplify presentation, we do not consider Byzantine failures, and we assume that crashed processes do not recover.

The analysis of a broadcast algorithm usually depends on more properties of the underlying system than only its size and composition, as well as on parameters of the algorithm itself. Henceforth, we will use the term *environment*, denoted $\mathcal{E}$, to refer to the set of relevant system properties and algorithm parameters. Environment $\mathcal{E}$ represents a point in an *environment space* $\mathbb{E}$, a set of all possible combinations of parameters: $\mathcal{E} \in \mathbb{E}$.

Let B_1 and B_2 be two broadcast algorithms that have different sets of parameters in their respective environments $\mathcal{E}_1$ and $\mathcal{E}_2$. To compare the algorithms we introduce a *compound* environment - a union of the two environments, $\mathcal{E} = \mathcal{E}_1 \cup \mathcal{E}_2$. Note that the composition makes sense only if the related parameters in $\mathcal{E}_1$ and $\mathcal{E}_2$ do not contradict. For example, if the system models for B_1 and B_2 comprise the probabilities of an end-to-end message loss, respectively, $\varepsilon_1 \in \mathcal{E}_1$ and $\varepsilon_2 \in \mathcal{E}_2$, then $\varepsilon_1 = \varepsilon_2$. Otherwise, the comparison does not seem meaningful. In Section 5 we will illustrate this through the concrete examples.

2.2. Δ-Reliable Broadcast

Let Δ be any pair of real numbers (ψ, ρ) $(\psi, \rho \in [0, 1])$. We say that a broadcast protocol *complies with the specification of* **Δ-Reliable Broadcast** (or a broadcast protocol *is* **Δ-Reliable**) iff the following properties are *simultaneously satisfied with probability* ψ:

Integrity For any message m, every correct process delivers m at most once, and only if m was previously broadcast by $sender(m)$.

Validity If a correct process p broadcasts a message m then p eventually delivers m.

Δ-Agreement If a correct process delivers a message m, then eventually at least a fraction ρ of correct processes deliver m.

Properties Validity and Integrity here are the same as in traditional Reliable Broadcast [6].

Agreement, as defined in [6], is transformed here into Δ-Agreement which is less restrictive in terms of the number of processes that need to deliver the message and also has a probabilistic flavor.

2.3. Interpretation of ρ and ψ

$\Delta = (\psi, \rho)$ represents a basic "reliability measure" of a broadcast algorithm. The values of ψ and ρ are intrinsically coupled: ψ can roughly be pictured as the probability with which at least a fraction ρ of processes behave according to the properties of Reliable Broadcast [6]:

Reliability probability ψ: ψ is the probability that a protocol run behaves "properly". That is, once a message m is broadcast by a correct process, "enough" correct processes eventually deliver m.

Reliability degree ρ: ρ defines the fraction of correct processes which eventually deliver m.

For instance, to satisfy the properties of Δ-Reliable Broadcast with $\Delta = (\psi = 0.95, \rho = 0.9)$, once a message m is broadcast, an algorithm should, with probability 0.95, deliver m to 90% of correct processes in the system. In other terms, in a run of the system with 10 correct processes, one can expect 95% of all messages which are broadcast to be delivered by at least 9 processes (not necessarily the same processes for every message).

2.4. Reliability distribution function

In a practical system, with a given required reliability degree ρ, several broadcast algorithms can easily be compared along the ψ they offer for the given ρ. To give an informal measure of the general performance in terms of reliability of a broadcast algorithm, several samples $\Delta_1 ... \Delta_s$ are usually sufficient. A precise expression of the reliability of such an algorithm requires however the consideration of the probabilities for all possible $\rho \in [0, 1]$, especially when comparing two algorithms in general. Indeed, consider two algorithms B_1 and B_2 and a set Δ_{B_1}=(0.9, 0.9) and Δ_{B_2}=(0.85, 0.9). Algorithm B_1 seems to perform better for $\rho_{B_1} = \rho_{B_2} = 0.9$. However, this information is not sufficient to promote algorithm B_1 as "more reliable" than algorithm B_2, since for $\rho'_{B_1} = \rho'_{B_2} = 0.95$, algorithm B_2 might offer a ψ'_{B_2} of 0.8, while in the case of algorithm B_1, ψ'_{B_1} might be only 0.7.

To compare two algorithms in a more general manner, we define a *reliability distribution function* $\psi_B(\rho, \mathcal{E})$ of a broadcast algorithm B:

$$\psi_B : [0, 1] \times \mathbb{E} \mapsto [0, 1] \tag{1}$$

such that for any $\rho \in [0, 1]$ and $\mathcal{E} \in \mathbb{E}$, B is Δ-Reliable with $\Delta = (\psi_B(\rho, \mathcal{E}), \rho)$.

As a direct consequence of the definition of Δ-Agreement — a sample in which a fraction ρ_0 of processes deliver every message is also a sample in which *at least* any fraction $\rho \in [0, \rho_0]$ of the processes deliver every message — $\psi(\rho)$ is a *monotonically decreasing* (with respect to ρ) function.

Note however, that by the size of "a fraction ρ of n processes" we mean $\lceil \rho n \rceil$. Accordingly, $\psi(\rho)$ is not represented by a continuous function, but manifests steps.

2.5. Comparing broadcast algorithms

Consider a *reliability range* $\nabla = [\rho_1, \rho_2] \subseteq [0, 1]$, that is, a range of values for the reliability degree ρ which is of interest in the context of a comparison.

In the sense of Δ-Reliable Broadcast, in the environment $\mathcal{E}$, an algorithm B_1 is *more reliable in* $\nabla = [\rho_1, \rho_2]$ than an algorithm B_2 iff

$$\begin{aligned} &\forall \rho \in \nabla : \ \psi_{B_1}(\rho, \mathcal{E}) \geq \psi_{B_2}(\rho, \mathcal{E}), \text{ and} \\ &\exists \rho_0 \in \nabla : \ \psi_{B_1}(\rho_0, \mathcal{E}) > \psi_{B_2}(\rho_0, \mathcal{E})^1 \end{aligned} \tag{2}$$

Similarly, in the environment $\mathcal{E}$, an algorithm B_1 is said to be *strictly more reliable in* $\nabla = [\rho_1, \rho_2]$ $(\rho_2 > 0)$ than an algorithm B_2 iff

$$\forall \rho \in \nabla, \ \rho \neq 0 : \ \psi_{B_1}(\rho, \mathcal{E}) > \psi_{B_2}(\rho, \mathcal{E}) \tag{3}$$

We exclude here $\rho = 0$, because for any broadcast algorithm B: $\psi_B(0) = 1$.

Finally, in the environment $\mathcal{E}$, an algorithm B_1 is *more reliable* than an algorithm B_2 iff, in $\mathcal{E}$, B_1 is *more reliable* than B_2 in $\nabla = [0, 1]$. Analogously, in the environment $\mathcal{E}$, an algorithm B_1 is *strictly more reliable* than an algorithm B_2 iff, in $\mathcal{E}$, B_1 is *strictly more reliable* than B_2 in $\nabla = [0, 1]$.

2.6. Atomicity

The reliability distribution function can be used to define the probability that a certain number of processes deliver the message as a result of an algorithm run. More precisely, the probability that the fraction ρ of correct processes that delivered a broadcast message (in a given environment $\mathcal{E}$) is larger than ρ_1 but smaller than ρ_2 $(0 \leq \rho_1 < \rho_2 \leq 1)$ can be defined as:

$$P(\rho_1 \leq \rho < \rho_2) = \psi(\rho_1, \mathcal{E}) - \psi(\rho_2, \mathcal{E}). \tag{4}$$

Thus, the following *Atomicity predicate* (see more examples in [2]) defines a *failed broadcast* to be one that reaches more than a fraction σ of correct processes, but less than a fraction $1 - \sigma$ of correct processes in a system $(\sigma < 1/2)$.

$$P(\sigma \leq \rho < 1 - \sigma) = \psi(\sigma, \mathcal{E}) - \psi(1 - \sigma, \mathcal{E}). \tag{5}$$

[4] discusses several alternative *non-binary* specifications of broadcast algorithms.

2.7. Δ-Reliable Broadcast: from perfect to useless

A reliability distribution function ψ in the sense of (1) can be found for *any* broadcast algorithm. We demonstrate this through the following extreme cases.

Dreamcast: One can easily see that an algorithm implementing traditional Reliable Broadcast [6] in a given environment $\mathcal{E}$ is Δ-Reliable with $\Delta = (1, 1)$. Since ψ_{RB} is a monotonically decreasing function, this sample univocally defines ψ_{RB}: $\forall \rho \in [0, 1]$ $\psi_{RB}(\rho, \mathcal{E}) =$

[1] This second condition is necessary to avoid that two equally performing algorithms are "each more reliable than the other".

1. One may call such an algorithm *perfectly* reliable. As we mentioned earlier in the introduction, its practical implementation in a network with unreliable processes and channels is expensive and not scalable.

Spellcast: A bogus algorithm which does nothing conforms to the specification of Δ-Reliable Broadcast such that $\forall \mathcal{E} \in \mathbb{E}$ with at least one correct process and $\forall \rho \in]0,1] : \psi_{UB}(\rho, \mathcal{E}) = 0$ ($\psi_{UB}(0, \mathcal{E}) = 1$).

3. Bimodal Multicast

This section focuses on the *Bimodal Multicast* [2] algorithm. While providing a lower reliability in terms of Δ-Reliability than a perfectly reliable protocol, it is in most cases more scalable and efficient. We first recall the algorithm, and then discuss its Δ-Reliability.

3.1. Protocol overview

Bimodal Multicast is composed of two subprotocols structured roughly as in the Internet MUSE protocol [7]. The first is an unreliable, hierarchical multicast (IP Multicast can be used where available) that makes best-effort attempt to deliver each message to its destination. The second is a two-phase *anti-entropy* protocol that operates in a series of asynchronous rounds. During each round, the first phase detects message losses; the second phase corrects such losses and executes only if needed.

For the analysis below, we use a simplified version of *the first phase of the anti-entropy protocol* of Bimodal Multicast, which differs from the original protocol in ways that simplify the discussion without changing the analytical results (also used by [2]). The algorithm proceeds as follows [2]. A message m which is gossiped about is attached the number of times it has been forwarded, $round$. When a process p receives m for the first time, p delivers it, and, if the message has been forwarded less than T rounds ($round < T$), forwards m to $n\beta$ randomly chosen processes by attaching it $round + 1$. When a process p broadcasts m, it handles m as if it had received m with 0 attached.

3.2. Model

The stochastic analysis below is based on the assumption that the execution of a broadcast algorithm can be broken up into a sequence of synchronous *rounds*, such that, during each round t, only processes which have gossips with round number t are gossiping, and every round happens strictly after all the transmission of the previous round are completed [1, 2]. Of course, in a real execution, each process autonomously proceeds in its own asynchronous rounds.

For the following analysis, we assume that failures are *stochastically independent*. The probability of a message loss is $\varepsilon > 0$, and the probability of a process crash during the protocol execution is $\tau > 0$. For simplicity, we assume that all incorrect processes are initially crashed. This implies that dependent link failures like a network partition are outside of our failure model. At any moment and for any message m, an *infected process* is one that already received m, an *infectious process* is an infected one which is gossiping m in the current round, and a *susceptible process* is one that is not infected yet by m. Following [2], we describe the state of the propagation of a given message n in round t using the random variables X_t, and Y_t, which denote the number of susceptible processes and the number of infectious processes, respectively.Initially, only the broadcastin process is infected. To summarize the constraints on the state of the system:

$$X_{t+1} + Y_{t+1} = X_t, \quad X_T + \sum_{t=0}^{T} Y_t = n. \tag{6}$$

with initial values $X_0 = n - 1, Y_0 = 1$. Note that at any round t, the number of infected processes is $n - X_t$.

3.3. Analysis

Let $F = f$ be the number of incorrect processes in a given run. We define $\beta(1-\varepsilon)(n-f)/n$ as the probability that a given gossip message m sent by an infectious process is successfully received by a given process p_i, that is: (a) the gossiping (infectious) process chooses p_i as destination, (b) message m is not lost in transmission, and (c), process p_i is correct. Respectively, $q_f = 1 - \beta(1-\varepsilon)(n-f)/n$ is the probability that a certain process did *not* receive a given gossip message from a particular infectious process. If j processes are gossiping in a given round, susceptible process p_i *is not infected* in this round with probability q_f^j.

The corresponding stochastic process can be expressed in the form of a *homogenous Markov chain* with a transition matrix defined by:

$$\begin{aligned} p_{ijklf} &= P(X_{t+1} = k, Y_{t+1} = l | X_t = i, Y_t = j, F = f) \\ &= \begin{cases} \binom{i}{l} \, (1 - q_f^j)^l q_f^{jk} & k + l = i \\ 0 & k + l \neq i \end{cases} \end{aligned} \tag{7}$$

The distribution of X_{t+1} and Y_{t+1} can be defined as:

$$\begin{aligned} &P(X_{t+1} = k, Y_{t+1} = l | F = f) \\ &\quad = \sum_i \sum_j P(X_t = i, Y_t = j | F = f) p_{ijklf} \end{aligned} \tag{8}$$

Using (6),(7) and (8), we can build a distribution of X_T and Y_T. We are interested in the probability that, for some

$\rho \in [0,1]$, not less than a fraction ρ of correct processes are infected up to round T:

$$\begin{aligned}
&\psi_{BM}(\rho, \mathcal{E}_{BM}) = \\
&= \sum_{f} P(F = f) P(X_T \leq n - \lceil \rho(n-f) \rceil | F = f) \\
&= \sum_{f} \binom{n}{f} (1-\tau)^{n-f} \tau^{f}. \qquad (9) \\
&\sum_{i \leq n - \lceil \rho(n-f) \rceil} \sum_{j} P(X_T = i, Y_T = j | F = f),
\end{aligned}$$

where $\mathcal{E}_{BM} = (\varepsilon, \tau, \mathrm{n}, \beta, \mathrm{T})$ is the set of system and algorithm parameters defining the current environment.

3.4. Δ-Reliability of Bimodal Multicast

Based on this, we formally characterize the Δ-Reliability of Bimodal Multicast [2].

Proposition 1 *For any environment $\mathcal{E}_{BM} = (\varepsilon, \tau, n, \beta, T)$ and any $\rho \in [0,1]$ Bimodal Multicast [2] is Δ-Reliable with $\Delta = (\psi_{BM}(\rho, \mathcal{E}_{BM}), \rho)$.*

Proof: Validity and Integrity follow directly from the algorithm description and the absence of Byzantine failures: the sender of a broadcast message delivers the message immediately and a process that receives the broadcast message delivers it only once. Thus, Validity and Integrity are always satisfied.

The proof of Δ-Agreement follows from the analysis above. Since $\psi_{BM}(\rho, \mathcal{E}_{BM})$ gives the probability of successfully infecting at least a fraction ρ of correct processes, given that initially one process is infected, Δ-Reliability with Δ=($\psi_{BM}(\rho, \mathcal{E}_{BM})$,$\rho$) is guaranteed.

3.5. Approximation of $\psi_{BM}(\rho, \mathcal{E}_{BM})$

Here we present a way to approximate the function $\psi_{BM}(\rho, \mathcal{E}_{BM})$. We describe the state of the system using the stochastic process $X(t)$ - the proportion of susceptible processes in round t.

Neglecting the fluctuation of $X(t)$ around its conditional expectation $x(t)$, we have the following *deterministic* approximation of the stochastic process:

$$x(t+1) = x(t) q^{n(x(t-1) - x(t))}, \qquad (10)$$

with the following initial conditions:

$$x(0) = \frac{n-1}{n}, \quad x(1) = x(0) q. \qquad (11)$$

Here $q = 1 - \beta(1-\varepsilon)(1-\tau)$. The approach is robust for large n, when the deviation of $X(t)$ is comparatively small [1]. We define the *average reliability degree* of the protocol $E_{BM}[\rho]$ as $1 - lim_{t \to +\infty} x(t)$. In other words, $E_{BM}[\rho]$ specifies the average fraction of the system infected by a broadcast message. From (10) we can derive the following relationship:

$$x(t+1) q^{nx(t)} = x(t) q^{nx(t-1)} = \frac{n-1}{n} q^{n}. \qquad (12)$$

Denote $q = 1 - \mu/n$, where $\mu = \beta n (1-\varepsilon)(1-\tau)$. Assume that βn is constant (the number of gossip messages sent by a process per round does not depend on the system size). For large n, $(1 - \mu/n)^n \approx e^{-\mu}$. Thus, we have the following recursive relationship:

$$x(t+1) = e^{\mu(x(t)-1)}, \quad x(0) = \frac{n-1}{n}. \qquad (13)$$

Note that, according to (13), $x(t)$ is a monotonically decreasing function. Applying Cauchy's theorem, we can approximate the discrete function $x(t)$ by a continuous one $y(t), t \in [0, +\infty[$, such that $x(t+1) - x(t) = \dot{y}(t)$ and $y(t) = x(t)$, $t \in \mathbb{N}$, yielding the following Cauchy problem:

$$y(0) = \frac{n-1}{n}, \quad \dot{y} = e^{\mu(y-1)} - y. \qquad (14)$$

The question is: what is the lower-bound asymptote of the susceptible fraction of the system $y(t)$ and how does it depend on n?

One can easily see that equation (14) does not depend on t and n, that is if $\varphi(t)$ is a solution of (14), then, for any t_0, $\varphi(t - t_0)$ is also a solution of (14). The system size n only impacts the initial condition (14). Thus the lower-bound asymptote of $y(t)$ does not depend on n: (14) defines the *time* necessary to approach it.

The lower-bound asymptote x_l can be roughly estimated for $y \ll 1$ through the following consideration:

$$e^{\mu(y-1)} = e^{-\mu} + e^{-\mu} \mu y + O(y^2) \Rightarrow x_l = \frac{e^{-\mu}}{1 - \mu e^{-\mu}} \qquad (15)$$

Assume that the maximal number of rounds T is sufficient to approach *closely* the upper bound of the infected fraction $1 - x_l$ (that is $T = O(\log n)$ [1]). Hence, we can approximate the probability that a given process is infected as a result of the run as $1 - x_l$. The reliability distribution function is approximated as:

$$\psi_{BM}(\rho, \mathcal{E}_{BM}) = \sum_{\lceil \rho n \rceil \leq i \leq n} \binom{n}{i} (1 - x_l)^i x_l^{n-i}. \qquad (16)$$

Note that we are approximating $\psi_{BM}(\rho, \mathcal{E}_{BM})$ by the probability that at least fraction ρ of *all* processes is infected. This is valid when $\tau \ll 1$. In general, (16) defines a *lower bound* on $\psi_{BM}(\rho, \mathcal{E}_{BM})$: the probability of infecting at least fraction ρ of *correct* processes can be only larger.

3.6. Average reliability of Bimodal Multicast

The above analysis allows to state the following result:

Proposition 2 *For any environment $\mathcal{E}_{BM} = (\varepsilon, \tau, n, \beta_n)$ in which $\tau \ll 1$, the average reliability degree $E_{BM}[\rho](n)$ as a function of system size n is such that:*

$$E_{BM}[\rho](n) \to_{n \to +\infty} 1 - \frac{e^{-\mu}}{1 - \mu e^{-\mu}}, \qquad (17)$$

where $\mu = \beta_n n(1-\varepsilon)(1-\tau)$.

The proof follows directly from the approximations presented above.

Note that if $\beta_n = \frac{k}{n}$ (such that the number of partners a process gossip to each round, $k = \beta_n n$ is constant), then the right-hand side of (17) is constant with respect to the system size. In other words, the expected reliability degree of Bimodal Multicast is stable with respect to the scale of the system. This a very valuable property for self-organizing systems, since for some *fixed* set of parameters of the algorithm, its reliability degree does not degrade as the system size increases. As we will see in the following section, IP Multicast is not scalable in this sense: its average reliability degree $E_{IPM}[\rho](n)$ decreases exponentially as n increases.

4. IP Multicast

In this section, we illustrate the notion of Δ-Reliable Broadcast through a second, in the traditional sense [6] inherently unreliable algorithm, namely IP Multicast [3].

4.1. Protocol overview

As its name reveals, IP Multicast is a so-called *network-level* datagram broadcast protocol directly based on IP. The transmission of such datagrams is not reliable, and basic IP Multicast does not consider message loss detection and reparation, making it inherently unreliable. In the context of IP Multicast, many different protocols have been described and deployed.

4.2. Model

We focus here on a sparse distribution of processes.

We suppose a spanning tree, as for instance the ones that are encountered with the *Protocol-Independent Multicast — Sparse Mode* (PIM-SM) [5] protocol, which is k-ary and of depth d. In other terms, we consider a regular spanning tree with a single (correct) broadcasting process located at the root, k^d receiving processes constituting the leaves of the tree, and every non-leaf node of the tree representing a router with k outgoing links. The system size is thus given by $n = k^d + 1 \approx k^d$, but we will consider n and k as parameters of the environment, and, since we are interested in large systems, we use $d = log_k n$. A spanning tree obtained in a real use case can always be captured by a possibly bigger spanning tree with a number of leaves of order n conforming to the above description.

Similarly to the analysis of Bimodal Multicast in the previous section, τ is the probability that a given process fails, and the probability of a message loss in a link between two nodes in the spanning tree as ε_l. In addition, we define as γ the probability of a router failure. We assume that all incorrect entities are initially crashed and the link failures are stochastically independent.

4.3. Analysis

Similarly to the analysis presented in the previous section, we propose a breakdown in successive rounds. These rounds however correspond to the levels in the spanning tree, that is, at round 1, the router of a broadcasting process forwards a given message m to the k routers representing its child nodes ($Y_0 = 1$). Due to failures, only $Y_1 \leq k$ will receive m. In any round $1 < t < d$, the Y_{t-1} “infectious” routers of level $t-1$ forward m to their kY_{t-1} child nodes (maximum of k^t). The probability p of a successful reception of m by an entity at level $t < d$ is therefore given by $p = (1-\gamma)(1-\varepsilon_l)$. At round $t = d$, the routers composing level $d-1$ finally send m to the processes constituting the leaves of the tree. We assume that $F = f$ processes are correct in a given run.

The probability of having a given number Y_t of “infected” entities at a given level $t > 0$ can be computed recursively based on the probabilities of any number of infected entities at level $t-1$. Finally, the probability of obtaining a given number of infected processes at the leaves of the spanning tree enables the computation of the fraction ρ of the correct processes in Π which have received m. For that end, we require the probability of having j infected entities at level $0 < t < d$ based on the number i of infected entities at the previous level:

$$p_{ij} = \binom{ik}{j} p^j (1-p)^{(ik-j)} \qquad (18)$$

Thus, the probability of having j infected entities at round $0 < t < d$ is given recursively by:

$$P(Y_t = j) = \sum_{0 \leq i \leq k^{t-1}} P(Y_{t-1} = i) p_{ij} \qquad (19)$$

Let $F = f$ be the number of incorrect processes in a given run. The probability p_d of successful transmission of message m from an infected router at level $d-1$ to a process at level d is given by $p_f = (1-\varepsilon_l)(n-f)/n$ and the probability of having j infected processes at level $t = d$ based on the number i of infected entities at the previous level:

$$p_{ijf} = \binom{ik}{j} p_f^j (1 - p_f)^{(ik-j)}. \qquad (20)$$

Thus, the probability of having j infected processes at round d is given recursively by:

$$P(Y_d = j | F = f) = \sum_{0 \leq i \leq k^{d-1}} P(Y_{d-1} = i) p_{ijf} \qquad (21)$$

As a direct consequence, the probability of having infected at least a fraction ρ of correct processes in a k-ary spanning tree of depth d is given by:

$$\begin{aligned} &\psi_{IPM}(\rho, \mathcal{E}_{IPM}) \\ &= \sum_f P(F = f) P(Y_d \geq \lceil \rho(n-f) \rceil | F = f) \\ &= \sum_f \binom{n}{f} (1 - \tau)^{n-f} \tau^f . \qquad (22) \\ &\sum_{\lceil \rho(n-f) \rceil \leq i \leq n-f} P(Y_d = j | F = f), \end{aligned}$$

where M is the number of correct processes in a given run and $\mathcal{E}_{IPM}$ is the environment defined as the set of parameters $\mathcal{E}_{IPM} = (\varepsilon_l, \tau, \gamma, n, k)$.

4.4. Δ-Reliability of IP Multicast

Based on (22), we are now able to formally characterize the Δ-Reliability of IP Multicast.

Proposition 3 *For any environment $\mathcal{E}_{IPM} = (\varepsilon_l, \tau, \gamma, n, k)$ and $\forall \rho \in [0, 1]$ IP Multicast is Δ-Reliable with $\Delta = (\psi_{IPM}(\rho, \mathcal{E}_{IPM}), \rho)$.*

Proof: The proof of Integrity follows from the semantics of IP and the absence of Byzantine failures, and Validity is assured with prevalent operating systems. Thus, Validity and Integrity are always satisfied in this model.

The proof of Δ-Agreement follows from the analysis above. $\psi_{IPM}(\rho, \mathcal{E}_{IPM})$ is equal to the probability of successfully infecting at least a fraction ρ of processes. Thus Δ-Reliability with Δ=($\psi_{IPM}(\rho, \mathcal{E}_{IPM})$,$\rho$) is guaranteed.

4.5. Average reliability degree of IP Multicast

The average fraction ρ of correct processes which receive m, $E_{IPM}[\rho]$, is given by:

$$E_{IPM}[\rho](n) = (1 - \varepsilon_l) p^{\log_k n - 1}. \qquad (23)$$

Furthermore, the probability that *all* n processes are correct and receive a given broadcast message m, $P(Y_d = n) = \psi(1)$, can be easily expressed in this model through:

$$P(Y_d = n) = p^{\frac{n-k}{k-1}} (1 - \varepsilon_l)^n (1 - \tau)^n \qquad (24)$$

5. Discussion

We present discuss here the reliability distribution functions (and also the average reliability degrees) of Bimodal Multicast and IP Multicast, which enable the *quantification* of the difference in (Δ-)reliability of these algorithms.Furthermore, we show that Bimodal Multicast, unlike IP Multicast, manifests no considerable reliability degradation as the system grows in size.

5.1. Environment

We assume that the system topology allows each process to maintain a k-ary spanning tree with d layers, whose leaves represent the other processes, and non-leaf members represent the routers. The probability of a message loss between processes used in the analysis of Bimodal Multicast (Section 3) is thus bounded by $\varepsilon = 1 - (1 - \varepsilon_l)^d (1 - \gamma)^{d-1}$, where ε_l is the probability of a message loss in a link between two corresponding nodes in the spanning tree and γ is the probability of a router failure. We consider Bimodal Multicast and IP Multicast in the compound environment $\mathcal{E}_R = \mathcal{E}_{BM} \cup \mathcal{E}_{IPM} = (\varepsilon_l = 0.05, \tau = 0.01, \gamma = 0.001, n = 256, k = 4, \beta = 0.02, T = 6)$. Non-common parameters of the environments $\mathcal{E}_{BM}$ and $\mathcal{E}_{IPM}$, β and T, are chosen in order to approach closely the upper-bound reliability degree $1 - x_l$ defined by (15).

5.2. Reliability distribution functions

Figure 1 presents the reliability distribution functions $\psi_{BM}(\rho, \mathcal{E}_R)$ of Bimodal Multicast and $\psi_{IPM}(\rho, \mathcal{E}_R)$ of IP Multicast in the "realistic" compound environment $\mathcal{E}_R$. As expected, $\psi_{BM}(\rho, \mathcal{E}_R) > \psi_{IPM}(\rho, \mathcal{E}_R) \forall \rho \in [0.55, 1]$, i.e., Bimodal Multicast is *strictly more reliable in* $\nabla = [0.55, 1]$ *in the environment* $\mathcal{E}_R$. However, in a "better" environment $\mathcal{E}_{R+}$ (with much smaller values for ε_l, τ and γ), IP Multicast may guarantee the same level of reliability as Bimodal Multicast. At the extremum, in a perfect environment $\mathcal{E}_P$ with $\varepsilon_l = \tau = \gamma = 0$, $\psi_{IPM}(\rho, \mathcal{E}_P) = 1 \forall \rho \in [0, 1]$. Bimodal Multicast on the other hand, even in the perfect system, admits the case when all the gossips of any given round are sent to already infected members and some part of the system will never get the broadcast message. Thus, $\forall \rho \in]0, 1[$, $\psi_{BM}(\rho, \mathcal{E}_P)$ is strictly less than (but can be arbitrarily close to) 1. This conveys the strong impact of the choice of the environment, in which two algorithms are to be compared, on the respective reliability distributions, and thus on the result of the comparison.

5.3. Scalability measure

Our reliability measure of broadcasts algorithm can be used to sketch a measure of the *scalability* of such algo-

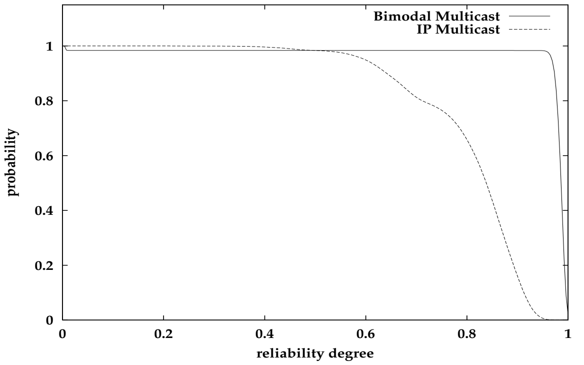

Figure 1. Reliability distribution functions $\psi_{BM}(\rho, \mathcal{E}_R)$ and $\psi_{IPM}(\rho, \mathcal{E}_R)$.

rithms. Basically, we can consider an algorithm to be scalable if its average reliability degree as function of the system size $E[\rho](n)$ is constant, or slowly decreasing.

This criterion obviously reflects just one dimension of scalability, namely that of reliability. Yet, investigating scalability in terms of overhead is not in the scope of this work. It is nevertheless worth noting that IP Multicast is "more scalable" in terms of message complexity and time: to obtain the same reliability degree it requests a smaller number of messages and consumes less time. Note furthermore that traditional Reliable Broadcast[6] is scalable in this context: its reliability degree is $E_{RB}[\rho](n) = 1$, although it is not scalable in terms of message complexity and time.

5.4. Average reliability degrees

Figure 2 presents the average reliability degrees for Bimodal Multicast and IP Multicast ($E_{BM}[\rho]$, resp. $E_{IPM}[\rho]$) for a varying sytem size n. As expected, n does not have a noticeable impact on the reliability of Bimodal Multicast (see Proposition 2) while, for IP Multicast, $E_{IPM}[\rho]$ is significantly decreasing.

6. Conclusions

This paper suggests a probabilistic *measure* of reliability, called Δ-*Reliability*. To demonstrate our measure, we considered the Bimodal Multicast algorithm of Birman et al. and a protocol variant of IP Multicast as case studies and we measured their respective reliabilities in probabilistic terms. The proposed specifications help to prove correctness of other probabilistic broadcast algorithms as well as to verify upper-layer distributed computing abstractions, which are based on reliable broadcast primitives such as Bimodal Multicast or IP Multicast.

To quantify the reliability of a broadcast algorithm in a probabilistic sense, we need the precise knowledge of system parameters and an accurate model of the behavior of the algorithm based on former ones. Such parameters are not always available, and models usually represent approximations. This outlines the main limitation of our notion of Δ-Reliable Broadcast: not every system model (and algorithm) matches it perfectly. We understand the notions we presented here as a first step towards defining a rigorous measure for scalable and probabilistic reliable protocols.

While the reliability offered by a broadcast algorithm can be quantified through our approach, there is no measure of its efficiency so far. We are thus currently working on identifying an appropriate measure for the efficiency, and maybe therethrough the scalability of broadcast algorithms.

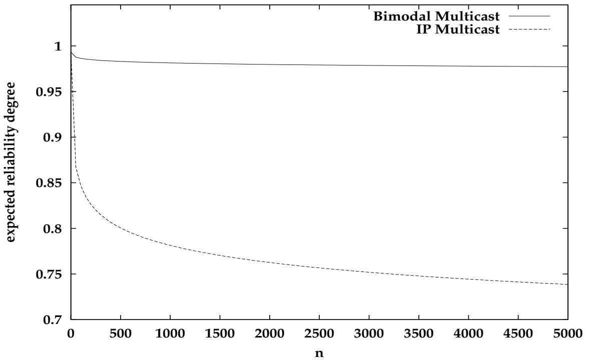

Figure 2. Average reliability degrees $E_{BM}[\rho]$ and $E_{IPM}[\rho]$ for a given system size n.

References

[1] N. Bailey. *The Mathematical Theory of Infectious Diseases and its Applications (second edition)*. Hafner Press, 1975.

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

[3] S. Deering and D. Cheriton. Multicast Routing in Datagram Internetworks and Extended LANs. *ACM Transactions on Computer Systems*, 8(2):85–110, May 1990.

[4] P. Eugster, P. Kouznetsov, and R. Guerraoui. $\Delta-$Reliable Broadcast. Technical report, Swiss Federal Institute of Technology, Lausanne, Jan. 2001.

[5] B. Fenner, M. Handley, H. Holbrook, and I. Kouvelas. Protocol Independent Multicast-Sparse Mode (PIM-SM): Protocol Specification. *Internet Engineering Task Force*, Nov. 2000.

[6] V. Hadzilacos and S. Toueg. Fault-Tolerant Broadcasts and Related Problems. In S. Mullender, *Distributed Systems*, chapter 5, pages 97–145. Addison-Wesley, 2nd ed., 1993.

[7] K. Lidl, J. Osborne, and J. Malcolm. Drinking from the firehose: Multicast USENET news. In *Proceedings of the Winter 1994 USENIX Conference*, pages 33–45, jan 1994.

[8] S. Paul, K. Sabnani, J. Lin, and S. Bhattacharyya. Reliable Multicast Transport Protocol (RMTP). *IEEE Journal on Selected Areas in Communications*, 15(3):407–421, Apr. 1997.