Epidemic Information Dissemination in Distributed Systems

Easy to deploy, robust, and highly resilient to failures, epidemic algorithms are a potentially effective mechanism for propagating information in large peer-to-peer systems deployed on Internet or ad hoc networks.

Patrick T. Eugster
Rachid Guerraoui
Swiss Federal Institute of Technology

Anne-Marie Kermarrec
Laurent Massoulié
Microsoft Research

The peer-to-peer (P2P) computing model offers a radically different and appealing alternative to the traditional client-server model for many large-scale applications in distributed settings. In this model, end-user processes share resources in a peer style, potentially acting as both client and server. The P2P approach removes central points of failure and associated performance bottlenecks; it also balances the load—such as forwarding messages or storing data—among all system processes, each of which requires only local knowledge of the system state. However, designing scalable P2P application-level protocols is not straightforward and remains an active area of research.

Epidemic algorithms have recently gained popularity as a potentially effective solution for disseminating information in large-scale systems,[1] particularly P2P systems deployed on Internet or ad hoc networks. In addition to their inherent scalability, they are easy to deploy, robust, and resilient to failure. It is possible to adjust the parameters of an epidemic algorithm to achieve high reliability despite process crashes and disconnections, packet losses, and a dynamic network topology.

Epidemic algorithms mimic the spread of a contagious disease. Just as infected individuals pass on a virus to those with whom they come into contact, each process in a distributed system relays new information it has received to randomly chosen peers rather than to a server or cluster of servers in charge of forwarding it. In turn, each of these processes forwards the information to other randomly selected processes, and so on.

As the "Mathematics of Epidemics" sidebars describe, much research has been devoted to observing, analyzing, and devising mathematical theories for epidemics. Once it has started, an epidemic is hard to eradicate: It only takes a few people to spread a disease, directly or indirectly, to the community at large. An epidemic is also highly resilient—even if many infected people die before they transmit the contagion or are immunized, the epidemic will reliably propagate throughout the population.

Although researchers have used epidemic algorithms in applications such as failure detection,[2] data aggregation, resource discovery and monitoring,[3] and database replication,[4] their general applicability to practical, Internet-wide systems remains open to question. We describe four key problems—membership maintenance, network awareness, buffer management, and message filtering—and suggest some preliminary approaches to address them.

DISSEMINATION PARAMETERS

In an epidemic algorithm, all system processes are potentially involved in information dissemination. Basically, every process buffers every message it receives up to a certain buffer capacity b and forwards that message a limited number of times t. The process forwards the message each time to a randomly selected set of processes of limited size f, the *fan-out* of the dissemination.

Many variants of epidemic algorithms exist and are typically distinguished by the values of b, t, and f. These parameters may be fixed independently of the number n of processes in the system, in which case the load imposed on every process remains

Published by the IEEE Computer Society
0018-9162/04/$20.00 © 2004 IEEE

bounded. The reliability of information delivery will then depend both on these values as well as on the system size. Alternatively, the dissemination parameters can evolve with n. In this case, reasonable load could be maintained if the parameters increase slowly with n—for example, logarithmically.

The inherent reliability of epidemic algorithms lies in a proactive mechanism that circumvents potential process and network link failures. As Figure 1 shows, every process that receives a message to be disseminated forwards it by default to a randomly chosen subset f of other processes. Each of these infected processes in turn forwards the information to another random subset. Thus, unlike reactive algorithms, in which processes react to failures by retransmitting missing information, epidemic algorithms do not require a mechanism to detect and reconfigure from failures.

In addition, epidemic algorithms exhibit *bimodal* behavior: They either achieve successful delivery to almost all processes or only reach a negligible portion of the processes. By tuning the protocol's parameters b, t, and f appropriately, epidemic algorithms can provide the same guarantees as deterministic algorithms.

Implementing an epidemic algorithm in a practical setting requires addressing specific design constraints that the system processes' resource requirements impose with respect to

- *membership*—how processes get to know each other, and how many they need to know;
- *network awareness*—how to make the connections among processes reflect the actual network topology to ensure acceptable performance;
- *buffer management*—which information to drop at a process when its storage buffer is full; and
- *message filtering*—how to take into account the actual interest of processes and decrease the probability that they receive and store information of no interest to them.

Although studies of natural epidemics can provide useful insights into these issues, innovative solutions are required because such studies have primarily focused on quenching epidemics rather than facilitating their spread, which is the goal of an epidemic algorithm.

MEMBERSHIP

Membership is a fundamental issue underlying deployment of epidemic algorithms. In an epidemic

Mathematics of Epidemics: Branching Processes

Lord Francis Galton, an explorer and anthropologist concerned with the survival of noble family names, pioneered the mathematical theory of epidemics in the second half of the 19th century. Galton introduced the *branching process*,[1] which became known as the Galton-Watson model after Reverend Henry William Watson obtained some early theoretical results.

According to this model, a given generation r has X_r individuals. Each individual of each generation gives birth, with some probability p_k, to k descendants, who will contribute to the next generation. Starting from a single individual at generation 1, the probability of extinction p_{ext} must satisfy

$$p_{ext} = \sum_{k \geq 1} p_k (p_{ext})^k .$$

Based on this implicit characterization, p_{ext} must equal 1 if the mean number of descendants per individual,

$$f = \sum_{k \geq 1} k p_k ,$$

is less than 1 while p_{ext} is less than one for $f > 1$, the exact value of p_{ext} depending on the specific probability weights $\{p_k\}$.

For example, if an individual gives birth to 0, 1, or 2 descendants with respective probabilities $(1 - p)^2$, $2p(1 - p)$, and p^2 for a given parameter p, the mean number of descendants per individual is $f = 2p$, and the above equation yields the explicit characterization that $p_{ext} = 1$ if $p \leq 1/2$, and $p_{ext} = (1/p - 1)^2$ if $p > 1/2$.

The Galton-Watson model exhibits *phase transition*: Continuously varying parameter f results in radically different behavior—namely, survival of the population or, in the epidemic context, ongoing propagation of the disease by infected individuals. Variants of this simple model incorporate temporal as well as spatial aspects and also distinguish between multiple types of individuals at each generation.

Reference

1. K.B. Athreya and P. Ney, *Branching Processes*, Springer-Verlag, 1972.

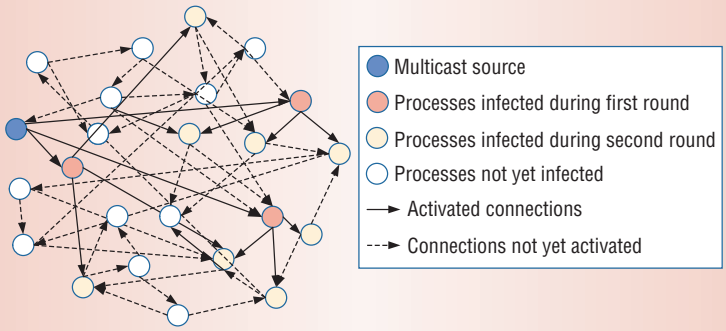

Figure 1. Epidemic algorithm. A multicast source, represented by the blue circle, sends a message to be disseminated in a system of size n. Each infected process—each process that receives the message—forwards it to a random subset of size $O(\log(n))$. Eventually, the message will reach all members of the system with a high probability after $O(\log(n))$ rounds. The failure of one or several communication links or processes does not significantly affect propagation of the message to live processes.

Mathematics of Epidemics: Finite Population Models

One modification of the basic branching process is incorporation of a population size n. X_r is now more naturally interpreted as the number of infectious individuals in the r-th round of epidemic spread. In each round, each such infectious individual will, with some probability p_k, try to contaminate k other members of the total population. These k members are chosen at random from the whole system.

Several variations of this model exist based on the number of rounds t that an individual remains infectious. The two extreme cases are the *infect and die* model, in which an individual tries to contaminate others for only one round and then stops, and the *infect forever* model, in which infected individuals remain infectious throughout.

A quantity of interest is the number Z_r of individuals infected prior to round r. Two key measures of the "success" of an epidemic dissemination are

- *proportion of infected processes*: the expected value of the fraction $Y_r = Z_r/n$ of the population infected after a given number of rounds r. The expectation of Y_r, that is, $E[Y_r] = E[Z_r]/n$, is the desired measure and represents the epidemic's success after a given amount of time.
- *probability of atomic infection*: the probability with which the entire population is infected after a given number of rounds, $P(Z_r = n)$. Informally, this value represents how likely the epidemic is to complete successfully after a given amount of time.

In the infect-forever model, assuming that infectious individuals try to contaminate f other members in each round, the approximate formula for the first measure—the expected fraction of infected members after r rounds[1]—is

$$Y_r \approx \frac{1}{1 + ne^{-fr}}.$$

Thus, the ratio of infected individuals to uninfected individuals increases exponentially—on average, by a factor of e^f in each round.

Reference

1. N.T.J. Bailey, *The Mathematical Theory of Infectious Diseases and Its Applications*, 2nd ed., Hafner Press, 1975.

dissemination, every process p that receives a message can forward it only to other processes that it knows. How a given process p acquires its own specific membership information impacts the performance of subsequent disseminations and is thus central to the design of scalable implementations of epidemic algorithms.

For example, the original epidemic broadcast algorithm[1] assumes that every process knows every other process—that is, every process has a list of all other processes in the system and therefore can communicate with them. This assumption is realistic assuming the epidemic broadcast scheme is deployed within a moderately sized cluster of processes. However, it becomes impractical when applied to large groups of processes because

- the storage required for the membership information increases linearly with the size of the system, and
- maintaining consistent views of the membership would impose an extra load on the network, particularly in a dynamic environment.

Examples of such dynamic environments are Internet P2P networks, in which processes can frequently flip between up and down states, and ad hoc networks, in which the quality of communication channels between processes can evolve quickly.

The scalability requirement thus imposes use of a decentralized protocol that provides each process with only a partial view of the system—that is, a subset of other processes' identities. An epidemic algorithm must therefore trade scalability against reliability: Small views growing sublinearly with the system size scale better, while large views reduce the probability that processes become isolated or that partitions occur.

One possible solution is to integrate membership with the epidemic dissemination itself: When a process forwards a message, it includes in this message a set of processes it knows; thus, the process that receives the message can update its list of known processes by adding new ones.[5] This approach alleviates the need for static membership without introducing new communication overhead.

Piggybacking membership information with regular message dissemination does not significantly increase message sizes because the added information is simply a list of process identifiers. A similar approach[6] relies on neighboring nodes periodically exchanging time-stamped messages and process identifiers and keeping only the most recent ones.

These partial membership approaches do, however, raise at least three issues.

- *Uniformity*. Every process in an epidemic algorithm forwards every message it receives to a subset of processes chosen uniformly at random among all processes in the system. Each process can make such a selection in a straightforward manner when it knows every other process; however, when only partial membership information is available, the process cannot make the selection unless the partial views of each process are themselves uniform samples of other processes, a property that is not trivial to ensure.
- *Adaptivity*. If the partial view size l and the dissemination parameters b, t, and f are predetermined and do not evolve as the system grows,

the probabilistic guarantees of delivery will vary with system size n. Therefore, to maintain a given probability of atomic broadcast, either the fan-out f or the latency t must increase with system size.[7] Increasing fan-out keeps latency constant, meaning that buffer size b does not have to adapt as significantly as it would to adapt to an increase in t. In any case, l and either t or f must adapt to system size, which presents a challenge because no individual process knows the precise value of n. Estimating system size in a fully decentralized way based on local knowledge remains an open problem.

- *Bootstrapping*. A closely related question is how processes initially get to know one another. This requires some external mechanism to initiate and trigger the dynamic membership scheme. Researchers must take such a mechanism into account when analyzing a dissemination scheme's probabilistic behavior.

In one approach that copes with these issues simultaneously, a new process joins the system by sending a join request to an arbitrary *contact* or bootstrapping process.[8] The newcomer then initializes its partial view with the contact process, which in turn propagates the request to all processes present in its own partial view. Each of these processes then either keeps the new process in its partial view or forwards the request to some process randomly chosen from its local view. This simple mechanism ensures that the system configures itself toward views of size $(c + 1) \log(n)$ on average, where c is a design parameter selected to ensure a high reliability for a target transmission failure probability.

The correct scaling of partial view lengths with system size depends critically on the contact process itself being chosen uniformly at random among existing processes. However, it is unlikely that the contact process initially reached is chosen at random; the expectation is that newcomers would contact one bootstrapping process among several with publicly advertised identities. An indirection mechanism based on weights reflecting the graph connectivity ensures that the contact process is effectively randomized even if all processes contact the same bootstrapping process to join a group.

NETWORK AWARENESS

Membership algorithms are oblivious to the underlying network topology and thus assume that all processes are equally reachable. It is therefore possible for a process to forward a message to a nearby process via a remote one. Consequently, these algorithms can impose a high load on the network, significantly limiting their applicability to Internet-wide settings.

Most solutions proposed to address this issue rely on a hierarchical organization of processes that attempts to reflect the network topology. The epi-

Mathematics of Epidemics: Proportion of Infected Processes

In the infect-and-die model, once infected, processes remain infectious for only one round before dying. Likewise, in an information dissemination system, each process will take action to communicate a message exactly once—namely, after receiving that message for the first time—but will not take further action, even when receiving subsequent copies of the same message.

For a large system size n, provided the epidemic catches, which occurs with probability $1 - p_{ext}$, the proportion of processes eventually contaminated, say π, satisfies the fixed-point equation, $\pi = 1 - e^{-\pi f}$, where f is the fan-out.

Because this equation does not rely on n, a fixed average number of descendants f will lead to the same proportion of eventually infected processs, π, irrespective of the system size provided this is large enough. This system exhibits the same type of phase transition as the basic branching scheme—namely, π becomes suddenly positive when f crosses the critical value 1. In addition, for a given value f, π is always smaller than 1, even though it approaches 1 as f increases.

Mathematics of Epidemics: Probability of Atomic Infection

In the infect-and-die model, for a fixed infection mechanism described by the probability weights p_k, the proportion π will actually be always smaller than 1. Thus, in large systems, although the probability that an arbitrary process will eventually receive the message that reads $(1 - p_{ext})\,\pi$ might be very large, the probability that all processes receive the message decreases to zero as the system size becomes large.

Atomic infection, or broadcast, characterizes an infection of all processes. The question arises of how to characterize system-size-dependent infection mechanisms, for which the probability that each process becomes infected is reasonably large.

Paul Erdös and Alfred Rényi, two Hungarian mathematicians, tackled this problem in the 1960s. Rather than viewing the evolutionary infection process, they examined the system's final state. This can be represented by a graph in which each node represents a system process. An arrow extends from a process m_1 to another process m_2 if m_1 has become infected and chooses to infect m_2. An epidemic started by member m_0 propagates to the whole system only if this graph contains a path from m_0 to any other process m.

Thus, the probability that all processes are infected is the probability that in a random graph there are paths from the originator m_0 to all other members. If the mean number of infected processes f evolves with the system size N, being equal to $\log(N) + c$ for some fixed parameter c, then the probability that the random graph is connected is given by

$$p_{connect} = e^{-e^{-c}}.$$

This phase transition from the state "not connected" to the state "connected" occurs when the key parameter $f/\log(n)$ crosses 1.

Mathematics of Epidemics: Latency of Infection

Consider how long it takes for a disease to reach every process in both the infect-and-die and infect-forever models.

In the infect-and-die model, the number f of targets for contamination must be of order $\log(n)$ for the infection to reach the whole system. Taking f to be the correct order, provided that the infection does indeed reach the entire system, Bella Bollobás[1] showed that the number of rounds R necessary to infect the entire system is

$$R = \frac{\log(n)}{\log(\log(n))} + O(1).$$

In the infect-forever model, assuming that each infectious process tries to contaminate f other processes in each round, Boris Pittel[2] showed that this number R satisfies

$$R = \log_{f+1}(n) + \frac{1}{f}\log(n) + O(1).$$

Thus in both models the epidemic spreads quickly, taking at most a logarithmic number of steps to reach every process.

References

1. B. Bollobás, *Random Graphs*, Cambridge Univ. Press, 2001.
2. B. Pittel, "On Spreading a Rumor," *SIAM J. Applied Mathematics*, vol. 47, no. 1, 1987, pp. 213-223.

demic algorithm then ensures that messages are mostly forwarded to processes within the same branch of the hierarchy, thereby limiting the load on core network routers. Only a few connections between subhierarchies are required to ensure successful implementation of epidemic dissemination.[7] However, organizing processes in a dynamic and fully distributed hierarchy is a challenging problem that continues to occupy researchers.

One possibility is to incorporate some form of administration service that is aware of the actual hierarchy.[3] This service assigns newly added processes to the hierarchy, which an epidemic algorithm can then exploit to limit network traffic. Another approach is to set up a two-level hierarchy in which processes favor the choice of low-connectivity neighbors as infection targets.[9] This technique aims to reduce the network overhead of epidemic algorithms when applied to wide area networks. The algorithm can weight infection targets probabilistically to favor close processes.

Yet another solution relies on a tree-like organization of processes that induces a hierarchy and provides each process with a membership that grows logarithmically with system size.[10] This presupposes that logical addresses associated with individual processes express information about the network topology, thus it is adaptable to any of the previous schemes. The tree underlying the algorithm is also used to selectively disseminate messages—that is, using a form of message filtering at each level of the tree, messages only propagate to subtrees hosting processes that are effectively interested in messages with such content.

In more complex mobile ad hoc networks, it is unlikely that a process knows or can even communicate with every other process. In manets, only processes on devices within a limited range can communicate directly, and indirect communication between two processes is only possible if they are connected through a chain of intermediate nodes. Network awareness is thus necessary to make communication not only more efficient, but also feasible. One way to address this issue is to have every process maintain a list of known processes as well as information on routes leading to those processes.[11]

BUFFER MANAGEMENT

Recall that in a simple epidemic broadcast algorithm, every process that receives a message must buffer it up to a certain capacity and forward it a limited number of times, each time to a randomly selected set of processes of limited size. Depending on the broadcast rate, a process's buffer capacity might be insufficient for it to forward every message it receives enough times to achieve acceptable reliability.

Directing a process to drop new messages when its buffer is full would prevent forwarding such messages. On the other hand, instructing a process to drop old messages when its buffer is full and new messages come in could result in some old messages not being forwarded a sufficient number of times. Researchers have considered two complementary approaches to deal with this problem.

Optimize memory usage

One approach assigns priorities to messages and, when the need for dropping messages arises, drops low-priority messages preferentially. Researchers have proposed at least two ways of defining priorities.

Age-based prioritization. A message's *age* is roughly equivalent to the number of times it has been transmitted. A process tags a message with its age before forwarding it to a new process. If a process's message buffer is full, it drops the oldest message—the message with the highest age—instead of dropping a message arbitrarily. Under certain conditions, this technique limits resource usage while preserving reliability.[5] Likewise, if a process must buffer some number of other process identities and its buffer is full, it can buffer lesser-known processes with higher priority.

Application semantics. Another way to define priorities is to rely on an application programmer to define an obsolescence relation between message pairs[12]: A process that receives message m_1 no longer needs message m_2 because, for example, m_2 contains information that subsumes m_2. It is possible to initially purge messages from a buffer using age-based prioritization and then, if necessary, remove yet more messages using application semantics.

Reduce information flow

Another way to ensure resource scalability while maintaining an acceptable degree of reliability is to reduce the flow of information the application produces. The challenge is to do so without introducing explicit feedback interactions between the producer of the information and processes with limited resources.

One solution is to exploit the epidemic flow itself, which requires every process to calculate the average buffer capability among all processes it communicates with and transmit that information. When the rate is too high with respect to that average, the process reduces that rate locally. Indirectly, the sources of the information get such feedback and reduce the rate of information production.

The main drawback with this approach is that the rate varies according to the process with the smallest buffer space. Designing alternative strategies that make better use of available buffer resources remains a challenging issue.

MESSAGE FILTERING

Ensuring that every message reaches every process in the system is the design objective when all processes are equally interested in receiving all messages. However, different groups of processes can have distinct interests. In this scenario, it might be desirable for the algorithm to first partition processes in the groups and then follow this objective for disseminating messages within each group.

An alternative approach is to enable processes within a single system to express specific interests and make sure they receive the appropriate messages—more precisely, to increase the probability P_1 that a process receives a message it is interested in and simultaneously decrease the probability P_2 that a process receives a message in which it is not interested.[10]

It is possible to enhance the epidemic dissemination scheme with filtering capabilities that trade complete randomization for some heuristic to inform interested processes in the dissemination about a given message. Nonrandomized solutions can store the complex interests and evaluate messages dynamically based on their contents to send them only to interested processes. However, in the context of scalable randomized algorithms, deploying an adequate filtering mechanism presents two basic problems:

- How does a process know which message is of interest to another process? Providing this knowledge in the system in a decentralized way is not trivial. Further, it is unclear how to integrate such information with the epidemic dissemination scheme itself.

Mathematics of Epidemics: The Small-World Phenomenon

At the other end of the spectrum from infection mechanisms in which processes choose contamination targets randomly from the total set of processes are spatially organized processes in which infections can only pass from neighbor to neighbor. In this case, the number of rounds needed to reach every process—previously logarithmic in system size n—is at least of order $\sqrt{n}$ for processes organized in a two-dimensional grid and, more generally, $n^{1/D}$ for a D-dimensional grid.

Between these two extremes is a model in which infectious processes transmit the disease to their neighbors as well as to a fixed number of known long-range contacts. Duncan Watts and Steven Strogatz[1] used this model to analyze the *small-world phenomenon*, focusing on long-range contacts chosen uniformly at random from the total set of processes. They presented a corresponding graph of infection transmissions demonstrating that introducing a single long-range contact per process is sufficient to dramatically modify the epidemic's behavior.[2,3] The number of rounds it takes to reach every process is of order $\log(n)$, as in the spatially unstructured case, even though the majority of disease transmissions are between neighbors.

David Kempe, Jon Kleinberg, and Alan Demers[4] recently studied how information could reach every process reasonably fast, as in the Watts-Strogatz model, but at the same time reach nearby processes much faster in the infect-forever scenario. They assumed that in each round an infectious process randomly picks a new process as a target for contamination and that each process u will choose a target v with a probability proportional to $d(u,v)^{-\rho D}$, where $d(u,v)$ is the distance between the two processes, D is the grid's dimension, and ρ is a positive coefficient strictly between 1 and 2.

In this setting, it is highly probable that an epidemic starting at process u will reach a process v within $\log^{1+\varepsilon}(d(u,v))$ for some positive parameter ε. That is, choosing long-range contacts based on some power of the distance to these contacts can bring further benefits to the original, uniform choice of long-range contacts.

References

1. D.J. Watts and S.H. Strogatz, "Collective Dynamics of 'Small-World' Networks," *Nature*, vol. 363, no. 6684, 1998, pp. 440-442.
2. M.E.J. Newman, C. Moore, and D.J. Watts, *Mean-Field Solution of the Small-World Network Model*, working paper 99-09-066, Santa Fe Inst., 1999.
3. A.D. Barbour and G. Reinert, "Small Worlds," *Random Structures and Algorithms*, vol. 19, no. 1, 2001, pp. 54-74.
4. D. Kempe, J.M. Kleinberg, and A.J. Demers, "Spatial Gossip and Resource Location Algorithms," *Proc. 33rd Ann. ACM Symp. Theory of Computing*, ACM Press, 2001, pp. 163-172.

- Even when a process p knows that a certain message is of no interest to another process q, does p unilaterally decide not to transmit the message to q? The answer initially would appear to be yes, to diminish probability P_2. However, this can impact probability P_1 because q might be critical in reaching other processes interested in the message.

If every process knows all other processes, they can route messages only to interested processes. However, when processes only know subsets of other processes in the system, the dissemination procedure's success depends on the quality of membership information. Making processes know and communicate mainly with processes manifesting similar interests is difficult, if not impossible, to achieve without a global knowledge of interests. In addition, desirable properties such as network awareness are even harder to accomplish with message filtering.

One approach that tries to find a compromise between the uncorrelated notions of physical and interest distance arranges processes hierarchically according to their geographical distances and groups their interests at each level in the hierarchy at the same time.[10] Informally, the algorithm relies on a two-level hierarchy. Each level combines the interests so that a process at any level only manifests the interests of all processes it represents recursively. This algorithm disseminates a message to all processes in the system in a number of rounds logarithmic in system size—similar to pure epidemic broadcast algorithms[1]—while only imposing membership knowledge of the logarithm of the system size on individual processes.

Implementing epidemic dissemination in a large-scale system requires connecting and managing the peers in a fully decentralized manner, thereby creating a peer-to-peer overlay network. Beyond the specific challenges we have discussed, a wider research agenda consists in extending the scope of epidemic algorithms from information dissemination to other applications that leverage the overlay network. Such applications would, for example, include content search, content-based publish/subscribe, and file sharing. ■

Acknowledgment

This work was partially funded by projects CH-FN-NCCR/PRN MICS IP 5.2, CH-FN 2100-064994.01/1, and IST CH-OFES No. 01.0227.

References

1. K.P. Birman et al., "Bimodal Multicast," *ACM Trans. Computer Systems,* vol. 17, no. 2, 1999, pp. 41-88.
2. R. van Renesse, Y. Minsky, and M. Hayden, "A Gossip-Style Failure Detection Service," *Middleware 98: IFIP Int'l Conf. Distributed Systems and Platforms and Open Distributed Processing,* N. Davies, K. Raymond, and J. Seitz, eds., Springer, 1998, pp. 55-70.
3. R. van Renesse, K.P. Birman, and W. Vogels, "Astrolabe: A Robust and Scalable Technology for Distributed Systems Monitoring, Management, and Data Mining," *ACM Trans. Computer Systems,* vol. 21, no. 2, 2003, pp. 164-206.
4. A.J. Demers et al., "Epidemic Algorithms for Replicated Database Maintenance," *Proc. 6th Ann. ACM Symp. Principles of Distributed Computing,* ACM Press, 1987, pp. 1-12.
5. P.T. Eugster et al., "Lightweight Probabilistic Broadcast," *ACM Trans. Computer Systems,* vol. 21, no. 4, 2003, pp. 341-374.
6. S. Voulgaris, M. Jelasity, and M. van Steen, "A Robust and Scalable Peer-to-Peer Gossiping Protocol," to appear in *Proc. 2nd Int'l Workshop Agents and Peer-to-Peer Computing,* LNCS 2872, Springer, 2003.
7. A-M. Kermarrec, L. Massoulié, and A.J. Ganesh, "Probabilistic Reliable Dissemination in Large-Scale Systems," *IEEE Trans. Parallel and Distributed Systems,* vol. 14, no. 3, 2003, pp. 248-258.
8. A.J. Ganesh, A-M. Kermarrec, and L. Massoulié, "Peer-to-Peer Membership Management for Gossip-Based Algorithms," *IEEE Trans. Computers,* vol. 52, no. 2, 2003, pp. 139-149.
9. M-J. Lin and K. Marzullo, *Directional Gossip: Gossip in a Wide Area Network,* tech. report CS1999-0622, Dept. Computer Science and Eng., Univ. of California, San Diego, 1999.
10. P.T. Eugster and R. Guerraoui, "Probabilistic Multicast," *Proc. Int'l Conf. Dependable Systems and Networks (DSN 02),* IEEE CS Press, 2002, pp. 313-324.
11. J. Luo, P.T. Eugster, and J-P. Hubaux, "Route-Driven Gossip: Probabilistic Reliable Multicast in Ad Hoc Networks," *Proc. 22nd Ann. Joint Conf. IEEE Computer and Comm. Societies,* IEEE CS Press, 2003, pp. 1-11.
12. J. Pereira, L. Rodrigues, and R. Oliveira, "Semantically Reliable Multicast: Definition, Implementation, and Performance Evaluation," *IEEE Trans. Computers,* vol. 52, no. 2, 2003, pp. 150-165.

Patrick T. Eugster *is a postdoctoral researcher in the School of Computer and Communication Sciences and the Distributed Programming Laboratory at the Swiss Federal Institute of Technology in Lausanne (EFPL). His research interests include distributed algorithms, fault tolerance, distributed*

programming, object-oriented programming, and middleware. Eugster received a PhD in computer science from EFPL. Contact him at Patrick.eugster@epfl.ch.

***Rachid Guerraoui** is a professor in the School of Computer and Communication Sciences and director of the Distributed Programming Laboratory at EFPL. His research interests include distributed algorithms, distributed systems, and object-oriented programming. Guerraoui received a PhD in computer science from the University of Orsay, France. He is a member of the IEEE and the ACM. Contact him at rachid.guerraoui@epfl.ch.*

***Anne-Marie Kermarrec** is a researcher at Microsoft Research Ltd. in Cambridge, United Kingdom. Her research interests include P2P computing and high-availability and application-level multicast in large-scale distributed systems. Kermarrec received a PhD in computer science from the University of Rennes, France. She is a member of the ACM. Contact her at annemk@microsoft.com.*

***Laurent Massoulié** is a member of the networking group at Microsoft Research Ltd. in Cambridge. His research interests are in the management of overlay networks for supporting P2P applications, congestion and admission control for data flows across the Internet, probabilistic modeling, and performance analysis of telecommunication systems. Massoulié received a PhD in automatic control from the University of Orsay, France. Contact him at lmassoul@microsoft.com.*