From Epidemics to Distributed Computing

P.T. Eugster, R. Guerraoui, A.-M. Kermarrec, L. Massoulié

***Abstract*— Epidemic algorithms have been recently recognized as robust and scalable means to disseminate information in large-scale settings. Information is disseminated reliably in a distributed system the same way an epidemic would be propagated throughout a group of individuals: each process of the system chooses random peers to whom it relays the information it has received. The underlying peer-to-peer communication paradigm is the key to the scalability of the dissemination scheme.**

Epidemic algorithms have been studied theoretically and their analysis is built on sound mathematical foundations. Although promising, their general applicability to large scale distributed systems has yet to go through addressing many issues. These constitute an exciting research agenda.

***Index Terms*— Scalability, peer-to-peer, epidemics, information dissemination, random graphs, probabilistic reliability.**

THE SERVER IS EVERYWHERE

The traditional client-server computing model is not adequate to address reliability and scalability properties. Even when servers are replicated for fault-tolerance, the synchronisation mechanism needed to preserve replica consistency is a major source of overhead.

The peer-to-peer computing model represents a radically different and a plausible alternative for many large scale applications in Internet-wide settings. In this model, resources are shared between end-user processes themselves in a peer style, meaning that every such process potentially acts both as a client and a server. Central points of failures disappear as well as the associated performance bottlenecks. Scalability is achieved because the load, whether it consists in forwarding messages or storing data for example, is balanced between all processes of the system. In addition, each process requires only local knowledge of the state of the system.

Application-level multicast is one of the distributed applications which may benefit from such a communication paradigm to scale to large systems. The design of scalable peer-to-peer application-level multicast protocols is not straightforward and represents an active area of research. This is one of the challenges that epidemic information dissemination algorithms take up [2].

EPIDEMIC INFORMATION DISSEMINATION

Epidemic algorithms have recently gained popularity as a robust and scalable way of propagating information in distributed systems [1]. A process that wishes to disseminate a new piece of information to the system does not send it to a server, or a cluster of servers, in charge of forwarding it, but rather to a set of other peer processes, chosen at random. In turn, each of these processes does the same, and also forwards the information to randomly selected processes, and so forth.

The principle underlying this information dissemination technique mimics the spread of *epidemics*, and we talk about *epidemic information dissemination*. Another close analogy is with the spread of a rumour among humans via gossiping and we also talk about *gossip dissemination*.

Once started, epidemics are hard to eradicate: a few people infected by a contagious disease are able to spread it, directly or indirectly, to a large population. Epidemics are resilient to *failures* in the infection process. That is, even if many infected people die before being able to transmit the disease, or are immunised, the epidemic is still *reliably* propagated over populations.

Epidemic dissemination algorithms are simple and easy to deploy. In addition to their attractive scalability promises, epidemic algorithms exhibit a very stable behaviour even in the presence of a high rate of link and/or process failures. There is no single point of failure and the reliability degrades gracefully with the number of failures. A large amount of research has been devoted to observing, analysing, and devising mathematical theories for epidemics.

Applying the epidemic idea to disseminate information among a large number of processes with a dynamic connection topology is thus very appealing [8], [9]. Not surprisingly, the use of epidemics algorithms has been explored in applications such as failure detection [14], data aggregation [7], resource discovery and monitoring [16] or database replication [2]. These experimentations have however revealed several non-trivial issues that need to be addressed before epidemic algorithms can be applied in practical Internet-wide settings.

We recall below the basics of epidemic information dissemination and discuss some of these issues. The detailed description of how such algorithms are applied in distributed applications, as well as a comparison with other application-level protocols, are out of the scope of this paper.

Dissemination parameters

In an epidemic algorithm, every process of the system is potentially involved in the dissemination. Basically, every process buffers every message (information unit) 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, called the *fanout* of the dissemination.

Many variants of epidemic dissemination algorithms exist and are typically distinguished by the value of the various parameters: 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 bounded. The reliability of information delivery will then depend both on these values as well as on the size n of the system. Alternatively, the dissemination parameters can evolve with the system size n. In this case, reasonable load could be maintained if the parameters increase slowly with n, e.g., logarithmically.

Inherently, the reliability of epidemic algorithms is based on a *pro-active* mechanism where redundancy and randomization are deployed in the first place to circumvent potential process failures and network link failures [6]. This is because, by default, every process chooses randomly a subset of size f of other processes to which the information is forwarded. A process that receives the information selects another subset and so forth (see Fig. 1). No mechanism is needed to detect and reconfigure from failures, unlike *reactive* algorithms where processes react to failures by retransmitting missing information. Ultimately, permanently failed processes will have to be removed from the system.

In some sense, epidemic algorithms exhibit a *bimodal* behaviour: there is a sharp threshold in the dissemination parameters' values for which a reliable delivery is ensured with a high probability. The probabilistic guarantee of message delivery is directly related to the value of the dissemination parameters. These parameters can be tuned so that with arbitrarily high probability, the algorithm meets the guarantees that deterministic algorithms would provide.

When it comes to implementing an epidemic dissemination algorithm in a practical setting, specific design constraints regarding the resource requirement from every process need to be taken care of. The questions to be addressed include: (1) How do processes get to know

Mathematics of epidemics: branching processes

The mathematical theory of epidemics was pioneered by Lord Francis Galton, in the second half of the nineteenth century, who was concerned with the survival of noble family names. He thus introduced the following model, now known as the Galton-Watson process, or simply branching process (see for instance [Ath72]): at generation r, there are X_r individuals (for his application, male descendants of a given family). Each individually gives birth, independently of the others, to k (male) descendants, with probability p_k, that will participate to generation $r+1$. Starting from a single individual at generation 1, Reverend Watson established that the probability of extinction p_{ext} must satisfy

$$p_{ext} = \sum_{k \geq 0} p_{ext}^k p_k.$$

Based on this implicit characterisation of p_{ext}, one can prove that p_{ext} must be 1 if the mean number of descendants per individual, $f := \sum_{k\geq 0} kp_k$ is less than one, while p_{ext} is strictly less than one for $f > 1$, the exact value of p_{ext} depending on the specific probability weights $\{p_k\}$. For instance, if birth can be given to 0, 1 or 2 individuals with respective probabilities $(1-p)^2$, $2p(1-p)$ and p^2 respectively, for a given parameter p, the mean number of descendants per individual is $f = 2p$ and the above equation yields the explicit characterisation that $p_{ext} = 1$ if $p \leq 1/2$, and $p_{ext} = (1/p-1)^2$ if $p > 1/2$. This simple model already exhibits an interesting feature known among physicists as a phase transition: by continuously varying the parameter f a radically different behaviour emerges, namely the survival of the population, or in the epidemic context, the ongoing propagation of the disease by infected individuals. Variants of this basic model incorporating temporal as well as spatial aspects, and also distinguishing between multiple types of individuals at each generation, have been thoroughly studied.

References

[Ath72] K.B. Athreya and P. Ney. *Branching Processes*. Springer-Verlag, New York, 1972.

Mathematics of epidemics: finite population models

A modification of the basic branching process that is important for our purpose is the incorporation of a population size, say 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 variants can be considered, for instance depending on the number of rounds t that an individual remains infectious. We discuss here only the two extreme cases: 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:

1) *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, i.e., $E[Y_r] = E[Z_r]/n$, is the desired measure here. This value represents *how successful* the epidemic will have been *after a given amount of time*.
2) *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, one has the following approximate formula for the first measure, i.e. the expected fraction of infected members after r rounds (cf. [Bai75]):

$$Y_r \approx \frac{1}{1 + n\ e^{-f\ r}}.$$

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

References

[Bai75] N.T.J. Bailey. *The Mathematical Theory of Infectious Diseases and its Applications (second edition)*. Hafner Pres, 1975.

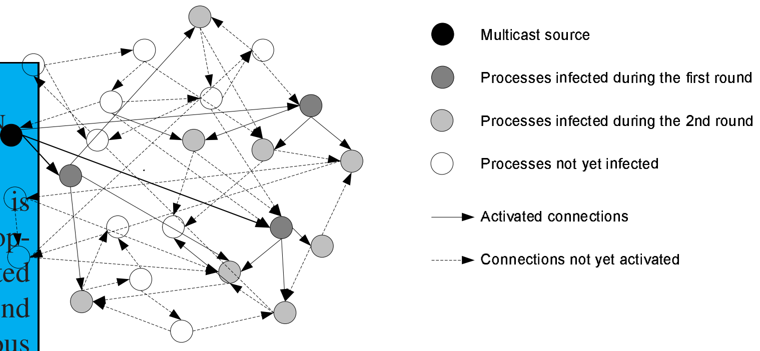

Fig. 1. **Illustration of an epidemic algorithm** A message to be disseminated in the system is first sent by the source, represented by the black circle. Each infected process (*i.e* each process having received the message) forwards the message to a random subset, of size $O(\log(n))$, n being the size of the system. 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 will hardly affect the propagation of the message to live processes.

each other, and how many do they need to know (*membership*)? (2) How to make the connections between processes reflect the actual network topology such that the performance is acceptable (*network awareness*)? (3) Which information to drop at a process when its storage buffer is full (*buffer management*)? (4) 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 (*message filtering*)?

Although studies of natural epidemics may provide useful insights in addressing these issues, the analogy goes only so far. The above issues call for innovative solutions because the knowledge derived from such studies has mainly been geared at quenching epidemics, whereas our concern here is rather the opposite, namely to allow for the epidemic to spread widely.

In the following, we discuss each of the four issues by summarizing some recent results and raising some open questions that, we believe, constitute an exciting research agenda.

Membership

Membership is a fundamental issue underlying the deployment of an epidemic information dissemination algorithm. Indeed, in an epidemic dissemination, every process p that receives a message may forward it only to other processes that it *knows*. It is then important to specify how any process p acquires its own specific membership information, as this will impact the performance of subsequent epidemic disseminations. This *who knows whom* relationship is probably the most important issue to consider while designing scalable implementations of epidemic algorithms.

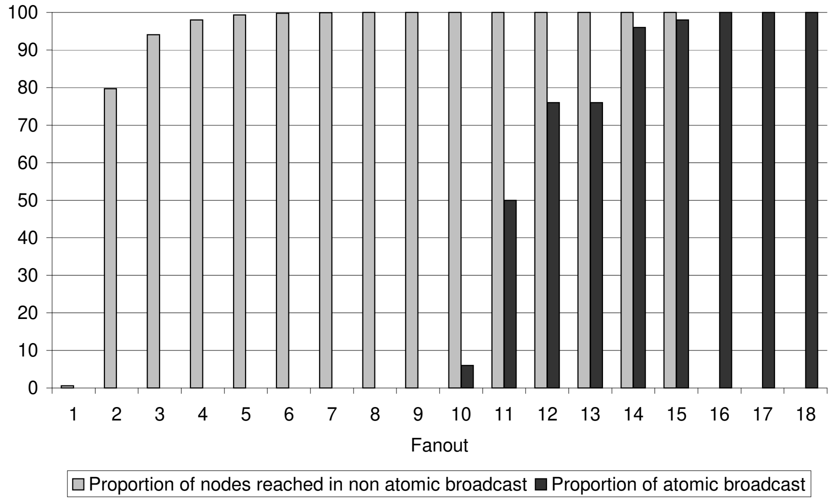

Fig. 2. **Epidemic dissemination in a 50,000 process group for different values of f.** Each infected process forwards the multicast message once to a fanout number f of other processes chosen uniformly at random. The dark bars denote the proportion of executions (results obtained out of 100 runs) where the broadcast was atomic (all processes were infected), while the light bars represent the proportion of processes that received the message in non-atomic broadcasts. Three different regions can be observed: *(i)* For low fanouts, atomicity is never achieved, and the proportion of reached processs increases from close to 0 to close to 1. *(ii)* For fanouts in an intermediate range, the proportion of reached processes stays close to 1, and the proportion of atomic broadcasts increases from 0 to 1. *(iii)* For high enough fanouts, almost all broadcasts are atomic.

In the original epidemic broadcast algorithm of [1] for instance, it is assumed that every process knows every other process: that is, every process has a list of all other processes in the system (and therefore is able to communicate with every such process). This assumption is realistic if we assume that the epidemic broadcast scheme is deployed within a moderately sized cluster of processes. It becomes unpractical when applied to large groups of processes for several reasons: first the storage required to store the membership information grows linearly with the size of the system. Secondly, maintaining consistent views of the membership would impose an extra load on the network, particularly in dynamic environments. Examples of such dynamic environments are peer-to-peer networks over the Internet, where processes may frequently flip between up and down states, or ad-hoc networks, where the quality of communication channels between processes may evolve quickly.

The requirement of scalability thus imposes to use a decentralized protocol providing each process only with a *partial view* of the system (that is, a subset of other processes' identities), on which an epidemic dissemination algorithm could rely. Such an algorithm must trade scalability against reliability: small views growing sublinearly with the system size obviously scale better, while large views reduce the probability that processes become isolated (lose any contact to the other processes), or that partitions occur.

One approach to this membership issue consists in mixing it with the epidemic dissemination itself [3]. The idea is the following: whenever a process forwards a message, it also includes in this message a set of processes it knows. Hence, the process that receives the message can enhance the list of processes it knew by adding new processes. This approach is attractive because it alleviates the need for the static membership assumption without introducing new communication overhead for exchanging membership information. The information is simply piggybacked with regular message dissemination. The message sizes are not significantly increased as the added information is simply a list of process identifiers.

A similar approach is described in [17]. It relies on neighboring nodes periodically exchanging time-stamped messages and process identifiers, and keeping only the most recent ones. The resulting who knows whom graphs have many properties in common with *small-worlds*.

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

1) *Uniformity.* The reliability of epidemic dissemination algorithms relies on the very fact that every process forwards every message it receives to a subset of processes chosen uniformly at random among *all* processes in the system. Such a uniform selection can be done by each process in a straightforward manner when it knows every other process. However, when only partial membership information is available, this can no longer be done unless the partial views of each process are

Mathematics of epidemics: Proportion of infected processes in the "infect and die" model

In this "infect and die" model, processes, once infected, remain infectious for one round precisely, before dying. In an information dissemination system, this reflects the fact that 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. The main result in this model is the following. For large system sizes n, provided the epidemic catches (which occurs with probability $1 - p_{ext}$), then the proportion of processes eventually contaminated, say π, satisfies the following fixed point equation

$$\pi = 1 - e^{-\pi f},$$

where f is the fanout introduced before. A remarkable feature of this equation is that it does not rely on the system size n, and thus 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. We observe in this system the same type of phase transition as in the basic branching scheme, namely that π becomes suddenly positive when f crosses the critical value 1. Another thing to note is that, for a given value f, the corresponding proportion of ultimately infected members, π, is always smaller than 1, even though it approaches 1 as f increases.

themselves uniform samples of other processes, a property that is not trivial to ensure.

2) *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 in size, the probabilistic guarantees of delivery will vary with the system size. To maintain a given probability of atomic broadcast in the face of an increasing system size, dissemination parameters must be adapted [10]: either the fanout f or the latency t need to increase with the system size. The advantage of increasing the fanout f is that the latency can be kept constant, meaning that also buffer sizes b do not have to adapt significantly, as would be the case when increasing the time for dissemination t. In any case, the view size l and either t or f need to be adapted to the system size. This constitutes a challenging issue, as one wants these parameters to adapt to the system size n,

Mathematics of epidemics: Probability of atomic infection

As we have just seen, in the "infect and die" model, for a fixed infection mechanism described by the p_k's, 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, which reads $(1 - p_{ext})\pi$, might be very large, the probability that all processs receive the message will go to zero as the system size becomes large. We refer to *atomic* infection (or broadcast) as characterizing an infection of *all* processes. The question then arises of how to characterise system size–dependent infection mechanisms, for which the probability that each process gets infected is reasonably large.

Such a question has been raised and successfully tackled in the 1960's by two famous Hungarian mathematicians, Paul Erdös and Alfred Renyi. Rather than viewing the evolutionary infection process, they instead looked directly at the final situation in which the system is left. This is best seen as a graph, where each node represents a process of the system, and one places an arrow from a process, say m_1, to another, say m_2, if m_1 would have chosen to infect m_2, had it been infected itself. An epidemic started by member m_0 propagates to the whole system if and only if in this graph, there exists a path from m_0 to any other process m. Thus the probability that all processes are infected is the probability that a random graph is connected. The main result on the connectivity of random graphs derived by Erdös and Renyi is as follows. 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, $p_{connect}$, is given by

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

Again, this is a phase transition, the transition from the state "not connected" to the state "connected" occurring when the key parameter $f/\log(n)$ crosses 1. Note that now the critical parameter $f/\log(n)$ depends on the system size. see Fig. 2 for an illustration.

Mathematics of epidemics: Latency of infection

Consider now how long it takes for the infection to reach every process, in both the "infect and die" and the "infect forever" models.

In the "infect forever" model, assuming that each infectious process tries to contaminate f other process in each round, it was shown by Boris Pittel in 1987 [Pit87] that the number of rounds R necessary to infect the entire system respects the following equation:

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

In the "infect and die" model, as we have seen, it is required that the number f of targets for contamination be of order $\log(n)$ for the infection to reach the whole system. Taking f to be of the correct order, provided that the infection does indeed reach the whole system, results of Bollobás (see [Bol01], p. 268) imply that the following holds:

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

Thus in both models, the epidemics do spread fast, in that it takes at most a logarithmic number of steps to reach everyone.

References

[Bol01] B. Bollobás. *Random Graphs*. Cambridge University Press, 2001.

[Pit87] B. Pittel. On spreading of a rumor. *SIAM Journal of Applied Mathematics*, 47:213–223, 1987.

while the precise value of n is not known to any individual process[1],

3) *Bootstrapping.* A closely related question is how processes get to know one another to start with? This requires some external mechanism to initiate and trigger the dynamic membership scheme. How to take into account such a mechanism when analysing the probabilistic behaviour of a dissemination scheme?

The approach proposed in [5] copes simultaneously with these issues. It consists for a new process to join the system by sending a join request to an arbitrary process, called a contact or a bootstrapping process. The newcomer will then initialize its partial view with the contact process. The latter process then propagates the request to all the 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

[1] Estimating the system size in a fully decentralized way based on local knowledge is still an open problem.

Mathematics of epidemics: Impact of spatial aspects, and the small world phenomenon

The preceding results applied to infection mechanisms that do not have any spatial structure: every target for contamination is chosen uniformly at random among the whole system.

At the other extreme, one might consider spatially organised processes (placed on a grid), and infections that can be passed from neighbour to neighbour only. This affects radically the behaviour of the epidemics. In particular, the number of rounds needed to reach everyone, previously logarithmic in the system size n, would now be at least of order $\sqrt{n}$ for a 2d-grid (and more generally, $n^{1/D}$ for a D-dimensional grid).

Between these two extremes, consider the following model: processes are spatially organised, say on a grid. An infectious process transmits the disease to its neighbors, and in addition to a fixed number of "long-range" contacts that it knows about. The corresponding graph of infection transmissions has been introduced by Watts and Strogatz [WS98], who studied more specifically the case where the long-range contacts are chosen uniformly at random from the total set of processes. One key result (see [NMW99] and [BR01]) is that the introduction of a single long-range contact per process is sufficient to modify dramatically the behaviour of the epidemics spread: the number of rounds it takes to reach everyone is then of order $\log(n)$, as in the spatially unstructured case, even though the majority of disease transmissions are between neighbours.

Watts and Strogatz considered the above model in order to analyse the so-called "small-world" phenomenon.

Recently, Kempe, Kleinberg and Demers [KKD01] have revisited this model to find a way of gossiping information so as to reach everyone reasonably fast, as in the Watts-Strogatz model, but at the same time to reach nearby processes much faster than in the Watts-Strogatz model. They actually consider the "infect forever" scenario, and assume that in each round an infectious process picks a new process at random as a target for contamination. They assume further that in each round, 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 dimension of the grid, and ρ is a positive coefficient strictly between 1 and 2. Their key reult is that in this setting, with high probability an epidemics started at a process u will reach a process v within $\log^{1+\epsilon}(d(u,v))$ 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

[BR01] A.D. Barbour and G. Reinert. Small worlds. *Random Structures and algorithms*, 19:54–74, 2001.

[KKD01] D. Kempe, J.M. Kleinberg, and A.J. Demers. Spatial gossip and resource location algorithms. In *33rd Annual ACM Syposium on Theory of Computing*, pages 163–172, 2001.

[NMW99] M.E.J. Newman, C. Moore, and D.J. Watts. Mean-field solution of the small-world network model. Technical Report 99-09-066, Santa Fe Institute, 1999.

[WS98] D. Watts and S. Strogatz. Collective dynamics of small-world networks. *Nature*, 1998.

simple mechanism ensures that the system configures itself towards 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 views lengths with system size depends critically on the fact that the contact process is itself chosen uniformly at random among existing processes. It is however unlikely that the contact process initially reached is chosen at random; one would rather expect newcomers to contact one bootstrapping process among a few whose identities are publicly advertised. An indirection mechanism based on weights reflecting the graph connectivity, ensures that even if the same bootstrapping process is contacted by all processes to join a group, the contact process is effectively randomized.

Network awareness

The membership algorithms discussed so far are oblivious to the underlying network topology and assume that all processes are equally reachable without taking into account that the impact on the network might vary widely. Therefore, it is perfectly possible that a message is forwarded from a process to a very close (in the sense of network topology) process via a remote one. As a result, these network oblivious algorithms may impose a high load on the network, which significantly limits their applicability to Internet-wide settings.

Most solutions proposed to address this issue rely on a hierarchical organisation of processes which attempts to reflect the network topology. The epidemic dissemination algorithm then ensures that messages are mostly forwarded to processes within the same branch of the hierarchy. This limits the load on core network routers. Only a few connections between sub-hierarchies are required to ensure successful implementation of epidemic dissemination [10]. Organizing processes into a hierarchy in a dynamic and fully distributed manner is however not straightforward and is still an active area of research.

The hierarchical organisation adopted in [16] relies on some form of administration system aware of the actual hierarchy. As they join, processes are assigned by this administration service to a hierarchy. An epidemic algorithm exploiting this hierarchy is then used to limit the network traffic. Another approach [11] consists in setting up a two-level hierarchy in which processes favour the choice of low connectivity neighbours as infection targets. This aims at reducing the network overhead of epidemic dissemination algorithms when applied to wide-area networks. Infection targets can actually be probabilistically weighted so that close processes are favoured.

The approach described in [4] relies on a tree-like organisation of processes, which induces a hierarchy, and provides each process with a membership that grows logarithmically with the system size. This presupposes logical addresses associated with individual processes expressing information about the network topology, and hence can be adapted to any of the previous schemes. The tree underlying the algorithm is also used to *selectively* disseminate messages, i.e., to perform a form of message filtering at each level of the tree in order to only propagate messages to subtrees hosting processes which are effectively interested in messages with such content. We shall return to this aspect later.

An additional level of complexity appears in mobile ad-hoc networks, where it is not only very unlikely that a process knows every other process, but also unlikely that a process may even communicate with every other (even known) process. Indeed, as direct communication is possible only between processes on devices within a limited communication range, and indirect communication between two processes is only possible if these happen to be connected through a chain of intermediate nodes. Network awareness is hence not only more required for communication to be efficient, but actually to be feasible. An attempt to address this issue is described in [12] where every process maintains a list of known processes, but also information on routes leading to those processes. The randomness of the partial views is given there by the randomness of the network process mobility pattern and the resulting feasible communication.

Buffer management

To understand the buffering problem, let us recall the principle of a simple epidemic broadcast algorithm: every process that receives a message (information unit) has to buffer that message up to a certain buffer capacity b, and to forward that message a limited number of times t, each time to a randomly selected set of processes of limited size f (fanout). Depending on the rate of new information production in the system (i.e., the broadcast rate), the buffer capacity of every process may be insufficient to ensure that every message is buffered long enough so that it can be forwarded a sufficient number of times to achieve an acceptable reliability.

Indeed, either one considers a conservative strategy where new messages are dropped when the buffer is full, and these new messages have no chance to be forwarded, or one considers a dynamic strategy where old messages are dropped whenever the buffer is full and new messages come in, but old messages might not be old enough to have been forwarded a sufficient number of times.

Two complementary approaches have been considered so far.

1) *Optimize memory usage.* This approach introduces some preferences in the treatment of messages. Priorities are introduced between messages and, when the need for dropping messages occurs, low priority messages are dropped preferentially. At least two ways of defining priorities have been suggested:
   - Messages are classified according to their *age*. Roughly speaking, the age of a message is the number of times the message has been transmitted. This notion is not local to a process but to a message: the age of a message is incremented whenever the message is transmitted to a new process, and the message is tagged with its age. If the message buffer of a process is full and the process has to drop a message, instead of dropping a message in an arbitrary way, the process chooses the oldest message, i.e., the message with the highest age. The conditions under which this technique enables us to limit the resources, and yet preserve reliability, are discussed in [3]. In fact, a similar idea can be applied to the partial membership information. If a process has to buffer a certain number of other process identities, and its buffer is full, it uses also a notion of age of a process. This notion captures the number of processes that know a given process. Processes that are less known are those buffered with higher priority.
   - Another way of defining priorities is by relying on application semantics [13]. The idea here is to assume that the programmer of the application has a way of relating pairs of messages by defining an *obsolescence* relation: message m_1 makes m_2 obsolete in the sense that a process that receives m_1 does not need m_2 anymore. For instance, m_1 contains information that subsumes m_2. Note that this idea can be combined with age-based priority. One would first choose the latter approach to purge messages from a buffer, then use the notion of age if messages still need to be purged. As a final alternative, one would still use randomization, or one of the conservative or dynamic approaches mentioned previously.
2) Another complementary approach to ensure resource scalability, while maintaining an acceptable degree of reliability, is by reducing the flow of information produced by the application. The idea is to influence the application by regulating its rate: when processes do not have enough resources, and enough time to buffer and forward messages a sufficient number of times, instead of increasing the resources, we decrease the rate of information flow. The challenge here is to do so without introducing explicit feedback interactions between the producer of the information and the processes with limited resources, as this would hamper the scalability of epidemic broadcast. An appealing idea is to exploit the epidemic flow itself to regulate the flow. The idea, explored in [15], 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 a feedback and themselves reduce the rate of information production. The main drawback here is that the rate is adjusted according to the process with the smallest buffer space. Designing alternative strategies which make better use of available buffer resources is a challenging issue.

Message filtering

So far, the stated design objective for epidemic dissemination algorithms was that every message should reach every process in the system. This is the desirable outcome when all processes are equally interested by all messages. In a scenario where distinct groups of processes have distinct interests, one could partition processes in the corresponding groups, and stick to this objective for disseminating messages within each group. An alternative approach is not to partition processes, but rather enable them, within a single system, to express specific interests, and make sure they receive messages they are interested in. More precisely, we would like to increase the probability P_1 that a process receives a message it is interested in, and at the same time decrease the probability P_2 that a process receives a message it is not interested in [4]. This goes through enhancing the epidemic dissemination scheme with *filtering* capabilities that trade the complete randomization with some heuristic to privilege interested processes in the dissemination of a given message. Non-randomized solutions exist which store the complex interests and evaluate dynamically messages based on their contents to send them to interested processes only. However, in the context of scalable randomized algorithms, the deployment of an adequate filtering mechanism is not trivial and at least two issues need to be addressed.

- 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. It is furthermore not clear how this can be integrated with the epidemic dissemination scheme itself.
- 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? At first glance, it might seem that the answer is yes in order to diminish probability P_2. But this can impact probability P_1 because q might be critical in reaching some other processes that are indeed interested in the message.

The tradeoff observed here can be viewed as a consequence of the brittleness of membership information. If every process knows all other processes, including their interests, messages can be routed to interested processes only. Whenever processes only know subsets of the other processes in the system however, the success of the dissemination procedure depends obviously on the quality of membership information. Clearly, making processes know and communicate mainly with processes manifesting similar interests is hard, if not impossible, to achieve without a global knowledge of interests. In addition, desirable properties such as network awareness are even more difficult to achieve with message filtering in mind. How can one find a compromise between the uncorrelated notions of physical and interest distance?

In the approach discussed in [4], processes are arranged in a form of hierarchy according to their geographical distances, while their interests are grouped at each level in the hierarchy at the same time. Informally, a subset of processes (level 1) are regrouped, and represented by a selected set of these processes (level 2) to the outside. Those prioritary processes are again regrouped with representatives of a limited number of subsets of level 1, of which again a subset are chosen to represent that subgroup of level 2, etc. At each level, the interests of processes are combined, such that a process at any level manifests interests of all processes it represents (recursively) only. This algorithm has the desirable property of completing a broadcast (i.e., the dissemination of a message to *all* processes in the system – the worst case) in a number of rounds logarithmic in the system size (similarly to pure epidemic broadcast algorithms, e.g., [1]), while only imposing a membership knowledge of the logarithm of the system size on individual processes.

SUMMARY

Epidemic algorithms are potentially effective solutions for disseminating information in large scale and dynamic systems. They are easy to deploy, robust and provide high resilience to failures. They proactively fight random process and network failures and do not need any reconfiguration when failures occur. This characteristic is particularly useful in peer-to-peer systems deployed on Internet or ad-hoc networks.

One can adjust the parameters of an epidemic information dissemination algorithm in such a way that a high reliability of dissemination is achieved despite process crashes and disconnections, packet losses and dynamic network topology. This provides the illusion of a global and virtual information system that every client can access and also takes part in implementing.

Although the idea is intuitive and appealing, putting it to work raises many questions. We described four of these issues: *membership maintenance, network awareness, buffer management* and *message filtering*. We have also reviewed some preliminary approaches to address these problems and raised some open questions. These questions, we believe, constitute an exciting research agenda.

REFERENCES

[1] 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, May 1999.

[2] A.J. Demers, D.H. Greene, C. Hauser, W. Irish, and J. Larson. Epidemic algorithms for replicated database maintenance. In *Proceedings of the Sixth Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 1–12, Vancouver, British Columbia, Canada, August 1987.

[3] P.T. Eugster, R. Guerraoui, S.B. Handurukande, A.-M. Kermarrec, and P. Kouznetsov. Lightweight probabilistic broadcast. *ACM Transaction on Computer Systems*, To appear, 2003.

[4] P.Th. Eugster and R. Guerraoui. Probabilistic multicast. In *IEEE International Conference on Dependable Systems and Networks (DSN 2002)*, 2002.

[5] A. Ganesh, A.-M. Kermarrec, and L. Massoulié. Peer-to-peer membership management for gossip-based algorithms. *IEEE Transactions on Computers*, 52(2), February 2003.

[6] I. Gupta, K.P. Birman, and R. van Renesse. Fighting fire with fire: using randomized gossip to combat stochastic scalability limits. *Quality and Reliability Engineering International*, 18:165–184, March 2002.

[7] I. Gupta, R. van Renesse, and K. Birman. Scalable fault-tolerant aggregation in large process groups, 2001.

[8] S.M. Hedetniemi, S.T. Hedetniemi, and A. Liestman. A survey of gossiping and broadcasting in communication networks. *Networks*, 1986.

[9] Richard M. Karp, Christian Schindelhauer, Scott Shenker, and Berthold Vocking. Randomized rumor spreading. In *IEEE Symposium on Foundations of Computer Science*, pages 565–574, 2000.

[10] A.-M. Kermarrec, L. Massoulié, and A.J. Ganesh. Probabilistic reliable dissemination in large-scale systems. *IEEE Transactions on Parallel and Distributed Systems*, 14(3), March 2003.

[11] M.-J. Lin and K. Marzullo. Directional gossip: Gossip in a wide-area network. Technical Report CS1999-0622, University of California, San Diego, Computer Science and Engineering, June 1999.

[12] J. Luo, P.Th. Eugster, and J.-P. Hubaux. Route driven gossip: Probabilistic reliable multicast in ad hoc networks. In *Infocom*, volume 39, 2003.

[13] J. Pereira, L. Rodrigues, and R. Oliveira. Semantically reliable multicast algorithms. *IEEE Transactions on Computers*, 52(2), February 2003.

[14] R. Van Renesse, Y.Minsky, and M. Hayden. A gossip-style failure detection service. In *IFIP Conference on Distributed Systems Platforms an Open Distributed Processing*, 1998.

[15] L. Rodrigues, S. Handurukande, J. Orlando, R. Guerraoui, and A.-M. Kermarrec. Adaptive gossip-based broadcast. In *IEEE International Conference on Dependable Systems and Networks (DSN)*, 2003.

[16] R. van Renesse, K. P. Birman, and W. Vogels. Astrolabe: A robust and scalable technology for distributed systems monitoring, management, and data mining. *ACM Transactions on Computer Systems*, 21(3), 2003.

[17] S. Voulgaris, M. Jelasity, and M. van Steen. A Robust and Scalable Peer-to-peer Gossiping Protocol In *2nd Int'l Workshop on Agents and Peer-to-Peer Computing (AP2PC 2003)*, 2003.