Scaling Up Publish/Subscribe Overlays Using Interest Correlation for Link Sharing

Miguel Matos, *Student Member*, *IEEE*, Pascal Felber, *Member*, *IEEE*, Rui Oliveira, *Member*, *IEEE*, José O. Pereira, and Etienne Rivière, *Member*, *IEEE*

Abstract—Topic-based publish/subscribe is at the core of many distributed systems, ranging from application integration middleware to news dissemination. Therefore, much research was dedicated to publish/subscribe architectures and protocols, and in particular to the design of overlay networks for decentralized topic-based routing and efficient message dissemination. Nonetheless, existing systems fail to take full advantage of shared interests when disseminating information, hence suffering from high maintenance and traffic costs, or construct overlays that cope poorly with the scale and dynamism of large networks. In this paper, we present StaN, a decentralized protocol that optimizes the properties of gossip-based overlay networks for topic-based publish/subscribe by sharing a large number of physical connections without disrupting its logical properties. StaN relies only on local knowledge and operates by leveraging common interests among participants to improve global resource usage and promote topic and event scalability. The experimental evaluation under two real workloads, both via a real deployment and through simulation, shows that StaN provides an attractive infrastructure for scalable topic-based publish/subscribe.

Index Terms—Publish and subscribe, scalability, topic-based, interest correlation, subscription clustering, link sharing

1 Introduction

1.1 Context

As society becomes ever more digital, the number of users connected to the Internet increases and the variety of data generated online grows steadily. Consequently, there is a huge demand in dissemination systems responsible for delivering data to their intended recipients in a variety of contexts, ranging from social networks and news sites to enterprise environments and financial markets.

The publish/subscribe paradigm emerged as an attractive model for scalable event dissemination, mainly due to the strong decoupling between the communicating entities: the producers (publishers) and consumers (subscribers) of information [1]. This flexibility—in terms of space, time, and synchronization—makes this model suitable to a wide range of application domains, from collaborative feed dissemination systems [2], [3] to enterprise service bus middleware used in service-oriented architectures.

The topic-based publish-subscribe variant categorizes items by explicit topics, avoiding the overhead of content-based filtering. Topics act as named channels where content can be published in the form of events. Participants interested in specific content subscribe to one or more topics and subsequently receive all the events published in these topics. Albeit simple, it captures the behavior of many real-world scenarios such as Usenet, Web syndication, Wikipedia, and social networks such as LiveJournal.

Topic-based publish/subscribe has attracted much interest among researchers (e.g., [4], [5], [6], [7], [8]), especially on the design of decentralized approaches that do not rely on a central broker to manage topics and route events. Furthermore, due to scale and dynamic behavior, many existing proposals rely on gossip-based dissemination [9], [10] to implement topic-based publish/subscribe communication [1].

Some designs build several stacked overlay networks, one for each topic, and have nodes independently join overlays for each of its subscriptions using traditional gossip-based protocols [9], [10], [11]. Unfortunately, this has high maintenance costs and presents scalability problems as the number of links established by each node grows linearly with the number of subscribed topics. Moreover, publishing the same event in multiple topics yields redundant transmissions, as events are separately disseminated among the same nodes through different overlays.

The alternative is to maintain a single overlay so that nodes with similar interests are close to each other [5], [12]. Shared interests are explicitly taken into account and redundant event transmissions on multiple topics avoided. With global knowledge of node interests, such semantic clustering can be achieved using gossip-based interactions [13], [14]. This has been formalized as the *minimum topic-connected* problem [7] and shown to be NP-complete. Furthermore, the resulting overlay is likely to exhibit a high clustering coefficient due to the approximation of nodes with similar interests. Therefore, it will be highly sensitive to faults and churn, and prone to partitioning [11].

1.2 Contributions

This paper presents StaN, a novel approach to topic-based publish-subscribe that aligns multiple independent overlays

• *M. Matos, R. Oliveira, and J.O. Pereira are with IHASLab - High-Assurance Software Lab, INESC TEC & Universidade do Minho, Campus de Gualtar, Braga 4710-057, Portugal.*
E-mail: miguelmatos@di.uminho.pt.
• *P. Felber and E. Rivière are with the Computer Science Department, University of Neuchâtel, Emile-Argand 11, CH-2000 Neuchatel, Switzerland.*

Manuscript received 7 Sept. 2011; revised 18 Nov. 2012; accepted 21 Dec. 2012; published online 10 Jan. 2013.
Recommended for acceptance by L. Xiao.
For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDS-2011-09-0600.
Digital Object Identifier no. 10.1109/TPDS.2013.6.

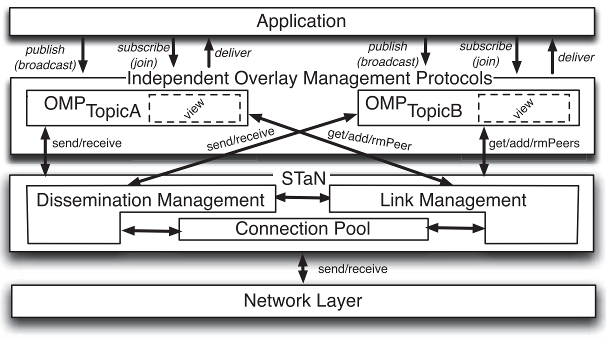


Fig. 1. StaN's architecture.

to promote link sharing among them. Despite being managed independently and in a decentralized fashion, provided subscription correlation, the overlays converge to share a large number of links. The growth is slower with the number of topics than traditional multioverlay approaches, thus promoting topic scalability. This is achieved while preserving the desirable properties for gossip-based dissemination, namely low-clustering coefficient and low diameter, thus making StaN an attractive infrastructure for efficient and scalable topic-based publish-subscribe.

Our contributions are the following: 1) a description of StaN detailing its operation and properties, 2) an evaluation on PlanetLab [15] and through simulations with a real workload from traces of LiveJournal [16] and Wikipedia [17], and 3) the design of a dissemination protocol that leverages StaN and messages published to multiple topics. The basic idea along with a preliminary evaluation based on synthetic workloads and simulations was presented in [18].

1.3 Roadmap

The remainder of this paper is organized as follows: We present StaN in Section 2 and evaluate it in Section 3. Related work is discussed in Section 4, and finally Section 5 concludes the paper. The dissemination protocol leveraging StaN and more evaluation results are presented in [19].

2 The StaN Protocol

In this section, we describe the system model, present StaN, and discuss its main properties.

2.1 System Model and Assumptions

To map as close as possible to real-world observations, we assume that topic popularity (the number of subscribers for a topic) and subscriptions per node (the number of topics a node is subscribed to) follow power-law distributions [20]. We also assume that topic subscriptions are correlated, i.e., there is a nonnegligible probability that subscription sets overlap as observed in real workloads [21], [22], [23]. We note that in the absence of correlation, StaN will degenerate in an overlay-per-topic solution. In this case, even single overlay approaches will produce disconnected components (one per topic). StaN's performance is therefore ultimately driven by the number of subscriptions per node and the correlation between topics.

StaN's architecture is presented in Fig. 1. Our approach assumes each topic has a separate random overlay, maintained by some overlay management protocol (OMP) like Scamp [24]. The key properties of these overlays are: 1) average view size grows logarithmically with the system size, thus enabling *node scalability*, and 2) clustering and diameter are low, making the overlays *fit* for gossip-based event dissemination and resilient in face of faults and churn [11]. Choosing peers uniformly at random is key to ensure those properties [25] and, therefore, it is fundamental to preserve randomness when combining links from different overlays. Link combination and alignment is done by the *Link Management* component.

We model an overlay as a directed graph and assume that the OMP maintains the graph connected. Using directed graphs allows each participant to make strictly local decisions regarding link establishment and removal. The set of neighbors of a node is called *view* and the number of neighbors is the *view size*.

Overlay links are a *logical* abstraction of the underlying network, which are mapped to a *physical link* by transport protocols such as TCP or UDP. This can be implemented by a dynamic pool of shared TCP/IP connections as in NeEM [26]. Therefore, a gossip-based dissemination protocol leveraging StaN can exploit logical links on different overlays that share the same physical link, thus avoiding redundant retransmission of the same event. This is achieved by the *Dissemination Management* component [19].

For simplicity, we do not differentiate publishers from subscribers and assume both are interested in receiving all events published on the topic. With this model of all-to-all communication, we use node as a means for both a publisher and a subscriber.

2.2 Design Rationale

When designing a topic-based publish/subscribe system, a set of desirable properties naturally emerge.

- *Completeness.* A node should receive all events published in the subscribed topics.
- *Accuracy.* A node should not receive any event from a topic it is not subscribed to.
- *Node scalability.* The system should scale with respect to the number of nodes for a given topic.
- *Topic scalability.* The system should scale with respect to the number of topics.
- *Fitness.* The overlays should have good structural properties to enable efficient event dissemination and resilience to faults and churn. These properties are *connectivity*, *low clustering coefficient*, and *low diameter* [11]. *Connectivity* indicates that all nodes are reachable from any other node and is fundamental to ensure completeness. *Clustering coefficient* measures the density of links among nodes. It is the ratio between the number of links that exist among the neighbors of a node and the total number of links that may exist, quantifying how close the neighbors are to being a clique. In practice, it is related to the dissemination cost, as highly clustered portions of the overlay will produce more redundant messages, and to fault tolerance, as highly clustered sections of the overlay tend to easily become disconnected from each other, thus compromising overall connectivity. *Diameter* gives a lower bound on the time and cost for a message to reach all nodes.

Proposals based on a single overlay, such as SpiderCast [5], recognize and exploit common subscriptions among nodes, allowing the number of links to grow sublinearly with the number of topics and nodes, thus providing *node scalability* and *topic scalability*. However, the remaining properties are more challenging to maintain. In particular, the *fitness* of the overlay degrades because semantic communities lead to high-clustering coefficients. *Accuracy* is also problematic as, eventually, nodes will have to relay events they are not interested in to guarantee *completeness*.

On the other hand, proposals like TERA [8] or daMulticast [27] that build one overlay per topic satisfy *completeness* and *accuracy* as each event is disseminated completely and only through the overlay it belongs to. *Fitness* and *node scalability* are also satisfied as these approaches rely on gossip-based OMPs designed for scalability and gossip-based dissemination (e.g., [10], [11]). The major drawback is *topic scalability* as the number of physical links established grows linearly with the number of topics each node subscribes to. Moreover, if an event matches more than one topic, these approaches cannot exploit this knowledge to reduce traffic because topics are fully separated.

StaN seeks to achieve the best of both worlds by addressing all aforementioned properties by combining several logical links in a single physical link. These combinations, detailed in Section 2.3, are strictly local decisions made by each node based on the set of other nodes in a topic-overlay. Link combination also allows an event published on multiple topics to be relayed just once through the *physical link* [19].

2.3 Link Management

The intuition behind StaN is very simple: each node periodically samples the subscribers of all its topics, that is, at each overlay it belongs to. Since, by assumption, subscriptions are correlated, these sets of sampled nodes will likely overlap. For each overlay, the node then deterministically selects a set of neighbors from the sampled nodes. This deterministic selection over overlapping sets leads, with high probability, to neighbors shared across all overlays enabling the mapping of several logical links to a single physical link, thus alleviating topic scalability problems.

Such design raises, however, two conflicting goals: first we want to promote link sharing by taking advantage of subscription correlation and second we do not want to induce clustering (due to subscription correlation) as this will impact the *fitness* of the overlay. The problem at hand is then to devise a neighbor selection process able to meet both goals. In the following, we study its key requirements.

To guarantee *fitness*, OMPs establish links *uniformly* at random [10], [11], [24], [25] and thus our neighbor selection process must preserve this randomness. Unfortunately, this implies that the probability that any two nodes are logically linked in more than one overlay is dismayingly small. Even with global knowledge of the system and full disclosure of the subscription sets, finding a minimal solution (with the smallest number of physical links) is NP-complete [7].

On the other hand, to promote link sharing, we need *determinism*, which is apparently conflicting with uniformly random choices. In fact, due to overlap in subscription sets resulting from correlation, a node using the same deterministic criterion in all overlays will independently choose approximately the same neighbors for each overlay.

To avoid clustering, nodes cannot choose the same set of neighbors (and neighbors of neighbors) which requires *asymmetry* in the local choices made by nodes.

In our approach, each node selects neighbors using a *pseudorandom* criterion that meets the above requirements. Uniformity is a key to preserve the good properties of a random overlay [25], while determinism is necessary to guarantee that a node will assign the same value to a target node independent of the overlay. Both are found in *hash functions* [28], which produce uniform outputs along its codomain and always map the same inputs to the same outputs. Thus, by feeding a hash function with the identifiers of known nodes, any node can obtain a uniform and deterministic sorting of all other nodes. Still, this is not sufficient as each node must have a different sort order; otherwise, the overlay would degrade into a chain-like structure. Besides sorting needs to be asymmetric to prevent clustering among neighbors. This is obtained by having each node supply a different input to the hash function, thus yielding different sorting orders. The pseudocode of our neighbor selection criterion (weight function) is shown in Algorithm 1 ($\mathrm{STR}(p)$ is the text representation of p's identifier).

Algorithm 1: Pseudo-random weight function on node p for target q

```
1 function WEIGHT($q$)
2     return HASH( STR($p$) + STR($q$))
```

The weight a node p assigns to a node q is given by the output of the hash function. Weights are assigned by concatenating p and q unique identifiers expressed as strings. Thus, nodes can locally order all other nodes and give preference to different sets of neighbors. Note finally that the weight function is asymmetric: considering any two nodes p and q, HASH(STR(p) + STR(q)) and Hash(STR(q) + STR(p)) yield different values with high probability [28].

The remaining challenge is to design a protocol that enables nodes to discover neighbors with minimum weight and replace links accordingly to reach the desired configuration. The asymmetry and absence of clustering precludes the use of well-known methods, such as T-Man [13], that rely on the establishment of a partial order among *all* nodes and dynamically converge the overlay toward a global target topology. As the weight function defines multiple orderings, one for each node in the system, there is no target topology and thus no such convergence guarantee.

Our proposal relies instead on random walks [29], a graph traversal procedure, to obtain uniform samples of the population. These samples are locally ordered according to the weight function and the neighbors for each overlay chosen accordingly.

The pseudocode for StaN is shown in Algorithm 2. *send* is a network-level primitive parameterized with the destination node's address and a message to be delivered at the receiver's side by means of the *receive* primitive. Each node p maintains one view per overlay t, denoted by

$p.views[t]$. The RANDOMNODE() function picks a random element from a list of nodes.

Algorithm 2: StaN protocol (node p)

```
    // Periodic refreshing of the views
1   periodically
2       foreach topic $t \in p.topics$ do
3           $q \leftarrow$ RANDOMNODE($p.views[t]$)
4           send COLLECTWALK($p, \varnothing$, TTL, $t$) to $q$

    // Random walk to collect nodes
5   upon receive COLLECTWALK($src, set, ttl, topic$)
6       $set \leftarrow set \cup \{p\}$
7       foreach node $n \in p.views[topic]$ do
8           $set \leftarrow set \cup \{n\}$
9       if $ttl > 0$ then
10          $q \leftarrow$ RANDOMNODE($p.views[topic]$)
11          send COLLECTWALK($src, set$, TTL $- 1, topic$) to $q$
12      else
13          send COLLECTREPLY($set, topic$) to $src$

    // Reply from last node in random walk
14  upon receive COLLECTREPLY($set, topic$)
15      $viewSize \leftarrow |p.views[topic]|$
16      $list \leftarrow \{q \in set \cup p.views[topic]$ sorted using WEIGHT($q$)$\}$
17      $p.views[topic] \leftarrow$ first $viewSize$ nodes from $list$
```

The protocol proceeds as follows: Periodically, each node initiates a random walk with a given *time-to-live* (TTL) in each overlay it belongs to. It selects a random node in that topic neighborhood and sends it a COLLECTWALK() message with its unique identifier, an empty set that will collect other nodes' identifiers, the desired TTL and the topic identifier (lines 1-4).

Upon reception of a COLLECTWALK() (lines 5-13), each node adds its own identifier and that of its neighbors to the received set, and forwards it to a random neighbor provided the TTL has not expired yet. Adding the neighbors to this set improves convergence time as more identifiers are collected by each random walk. When the TTL expires, the random walk ends and the node sends a COLLECTREPLY() with the set of identifiers back to the originator node.

Upon reception of this set (lines 14-17), the node computes its view size, merges the collected set with its own view, sorts the elements according to their weight, and finally selects the best nodes to replace its existing view without changing its size.

A simplified run of StaN with five nodes and two topics is depicted in Fig. 2. Node n_0 is subscribed to topics A and B and initially maintains four logical links, two for each topic, to neighbors n_1-n_4 (top figure). As there is no overlap in logical links, the number of physical links is also four. When running StaN, n_0 collects the ids of neighbors $n_1 - n_4$ and assigns then the weights, as shown in the figure. Then, n_0 replaces links to higher weight nodes with links to lower weight ones, on a per-overlay basis. Because of correlation, the logical links of overlays A and B converge to the same physical links (bottom figure): logical links are preserved and the number of physical links is reduced. Due to asymmetry, the weight n_0 assigns to, say, n_2 is unrelated to the weight n_2 assigns to n_0. This is reflected in the figure by not having n_2 and n_3 choose n_0 as its neighbor, nor each other.

2.4 Discussion

The weight function allows a node to order all other nodes *uniformly* at random, thus preserving the properties of the

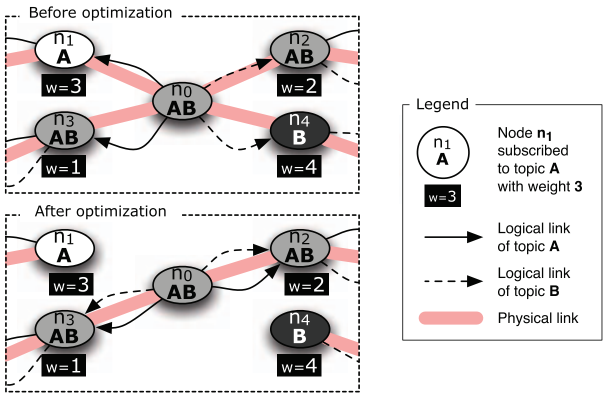


Fig. 2. 5-nodes sample run with two topics from the point of view of node n_0 (only a subset of the links is shown).

overlay and consequently its fitness. Moreover, it provides an *asymmetric* ordering of the identifier space, thus avoiding clustering. This is also reflected in the in- and out-view distributions as, overall, every node can be selected by another node with the same probability. Our method is substantially different from the node distance as used in consistent hashing [30], which clusters nodes according to their identifiers. As nodes tend to choose the same neighbors across overlays, due to *determinism*, logical links are often mapped to a single physical link, thus promoting topic scalability. This is because fewer physical links lead to improved resource usage in the form of open connections/sockets and protocol overhead, such as keep-alive messages. Completeness and node scalability are guaranteed by the underlying OMP (e.g., [11], [24]). Accuracy is satisfied by design as each topic uses a separate overlay.

StaN's overhead is due to the periodic random walks. This impact is low because the TTL is small and messages only carry a small sample of node ids. The TTL only needs to be of the order of the overlay diameter to provide the chance of discovering all nodes. Shorter TTLs would preclude nodes from opposite fringes of the overlay to know each other. Nodes can control the random walk period based on the expected improvements: new nodes would use small periods to quickly converge and then progressively reduce the frequency as improvements become marginal. Although not considered, nodes can also leverage other node's random walks (upon a COLLECTWALK()) as a source of new neighbors, further reducing the impact on the network and improving convergence speed. This optimization is also useful to counter the effects of message loss and churn, as nodes are able to gather more information for each message delivered.

3 Evaluation

We evaluate StaN using synthetic and real workloads by simulation and via a real deployment on PlanetLab. The evaluation is focused on performance and fitness. By evaluating the performance of StaN, we are able to infer its ability to promote link sharing among overlays which is fundamental to alleviate resource consumption in the form of physical links established, and thus promote topic scalability. By observing the impact StaN has on the properties that make overlays desirable for gossip-based

TABLE 1
Universe Configurations

LiveJournal

Name	$\mathcal{L}_0$	$\mathcal{L}_1$	$\mathcal{L}_2$	$\mathcal{L}_3$	$\mathcal{L}_4$	$\mathcal{L}_5$	$\mathcal{L}_6$	$\mathcal{L}_7$	$\mathcal{L}_8$
Seeds	all	all	20,000	15,000	10,000	5,000	1,000	500	100
Topics	28,904	13,652	13,129	12,608	11,674	9,331	3,215	1,805	253
Nodes	301,315	267,230	237,612	214,642	182,828	130,577	40,407	23,657	4,689

Wikipedia

Name	$\mathcal{W}_0$	$\mathcal{W}_1$	$\mathcal{W}_2$	$\mathcal{W}_3$	$\mathcal{W}_4$	$\mathcal{W}_5$	$\mathcal{W}_6$	$\mathcal{W}_7$	$\mathcal{W}_8$
Seeds	all	100,000	50,000	20,000	10,000	5,000	1,000	100	20
Topics	715,710	328,145	245,977	162,448	114,646	77,338	26,525	3,858	576
Nodes	2,015,060	761,225	497,854	277,474	175,152	108,620	33,956	5,957	1,381

dissemination, we assess StaN's suitability at achieving its performance goal. Evaluation under message loss and churn and a comparison of StaN with a global-omniscient approach are presented in a supplemental document, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPDS.2013.6, [19].

3.1 Experimental Data

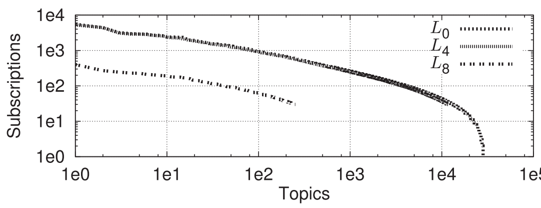
We used two real-world workloads: a trace of RSS subscriptions from LiveJournal [16] and a trace of edits of the English version of Wikipedia [17]. LiveJournal is a social network where users have a journal/blog in which they publish entries and can follow (subscribe to) the journals of others. The data gathered include the list of users and of subscribers to the journals. This collection of users and journals, with 28,904 journals and 301,315 users, forms the complete universe of our experiments. Journals map to topics and users to nodes subscribing those topics. For Wikipedia, we gathered the pages and page edits done by registered users until April 2012 resulting in 715,710 pages and 2,015,060 users. Pages map to topics and users who edited those pages to nodes subscribing those topics.

To increase the tractability of the universes and decrease experiment running time, we created smaller self-contained universes. A self-contained universe is created by selecting a random subset of topics, the seed set. We then select the users subscribed to topics in the seed set and add to the seed set the users' topic (journal) in LiveJournal's case or a random topic from the node's subscriptions in Wikipedia's case. The users subscribed to topics in the seed set comprise our universe pruning topics with less than 30 subscribers. As the self-contained universes were built using a random set of topics, the properties of subscription distribution are preserved. The complete generation method can be found in [4]. Table 1 describes all the LiveJournal and Wikipedia universes considered in the experiments. These workloads have been chosen as representatives of publish/subscribe systems with several seed set sizes. For each seed set size, we generated 100 universes, computed the ratio between the number of topics and nodes, and picked the median universe. Note that $\mathcal{L}_0$ and $\mathcal{W}_0$ represent the whole LiveJournal and Wikipedia universes, respectively.

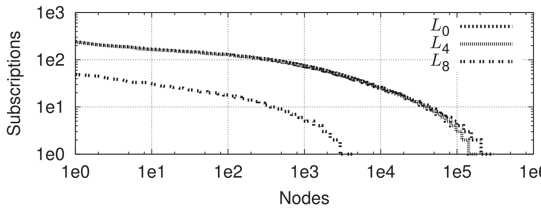
3.2 Workload Characteristics

We start by confirming that our assumptions about subscriptions distribution [20], [31] and correlation [21], [22], [23] hold. These assumptions are: 1) the number of subscribers to a given topic follows a power-law, 2) the number of subscriptions of each user follows a power-law, and 3) subscriptions are correlated with a nonnegligible probability.

Figs. 3 and 4 depict the distribution of subscriptions per topic (top) and subscriptions per node (bottom) for several LiveJournal and Wikipedia universes, respectively. Note that both plots are log-log. The general shape for both LiveJournal and Wikipedia is similar: few topics are highly popular while the vast majority has few subscribers, and

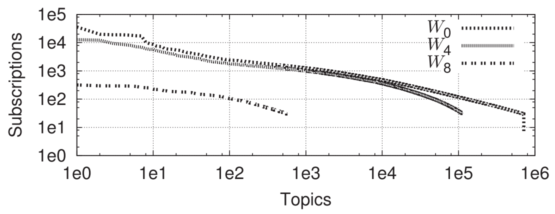


(a) Subscriptions per topic.

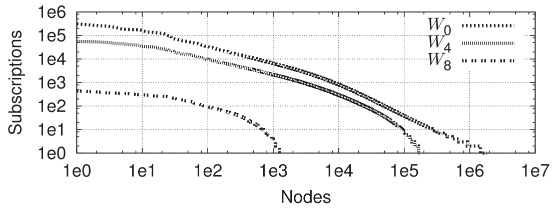


(b) Subscriptions per node.

Fig. 3. Subscription distribution for LiveJournal universes.



(a) Subscriptions per topic.



(b) Subscriptions per node.

Fig. 4. Subscription distribution for Wikipedia universes.

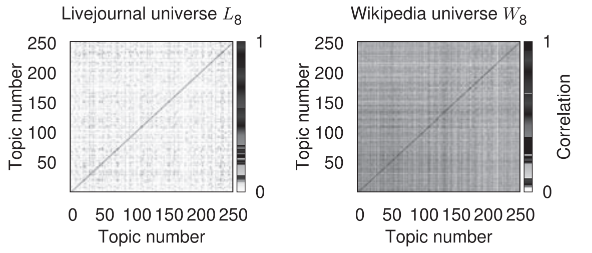


Fig. 5. Subscription correlation.

some users are subscribed to many topics while most subscribe to far fewer topics. These results confirm our assumptions about the subscription distribution [20], [31] and validate our method for the generation of smaller universes. Users subscribed to many topics are the ones that can encounter problems with topic scalability, as the number of physical connections they need to maintain can be quite large. StaN is therefore expected to mainly affect these nodes.

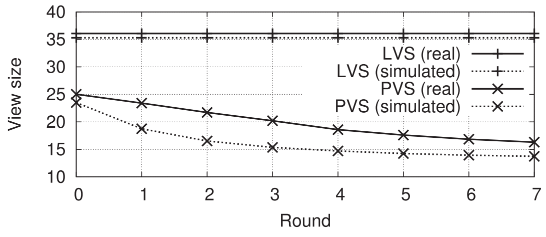
Fig. 5 depicts the correlation among subscriptions as a heat map, where white means no correlation and black means strong correlation. It was obtained by creating a matrix with topics as columns and subscribers as rows. For each subscriber, we set the value 1 in the respective column to indicate a subscription to the given topic, or 0 otherwise. We then calculate the Pearson correlation of the resulting matrix and plot it by mapping the values to different shades of gray. For the LiveJournal universe (left), there is a mild correlation among all topics (the map contains a nonnegligible amount of gray points), while for the Wikipedia universe (right) the correlation is stronger. This indicates that StaN should be able to promote physical link sharing on both universes, but to a greater extent on Wikipedia.

Finally, we devised a synthetic workload that provides finer control on the number of topics/nodes in a given universe to cope with the limitations of the PlanetLab [15] testbed. To this end, we built a two-dimensional grid with randomly placed node and topic identifiers. Additionally, each node is assigned an interest radius, and subscribes to topics whose identifiers fall within. For each topic, we randomly place several topic identifiers on the grid. The number of topic identifiers follows a power-law, thus matching the topic popularity model. The assigned interest radius also follows a power-law thus matching the node subscription model. Nodes close on the grid are likely to subscribe to the same topics, hence modeling subscription correlation. This synthetic workload closely matches our model and exhibits distributions similar to those observed on real universes [18].

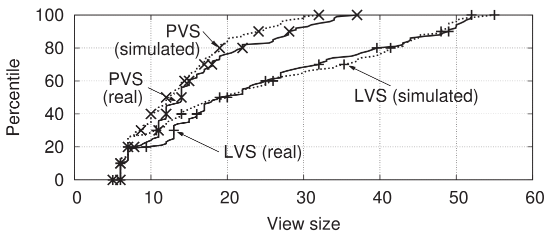
3.3 Experimental Setup

We evaluate StaN both through a real deployment on PlanetLab and by simulation. We could not perform all experiments on a real deployment due to scalability and resource limitations of PlanetLab that prevents experiments with bigger real universes. The deployment on PlanetLab is done with Splay [32], a framework for the development, deployment, and evaluation of distributed applications.[1]

1. Splay code used in the experiments is available at http://static.lsd.di.uminho.pt/stanTPDS.tbz2.



(a) Evolution.



(b) Distribution.

Fig. 6. Evolution and distribution of the LVS and PVS for a synthetic universe (100 nodes and 16 topics). The legend is shared by the two graphs.

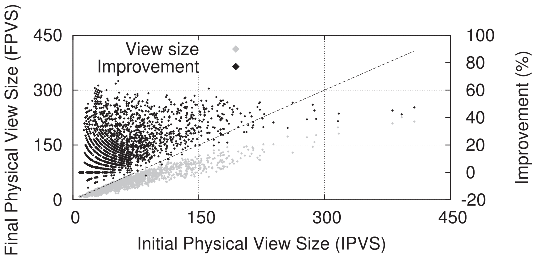
We also developed a discrete, round-based event simulator that can scale to thousands of nodes and topics to evaluate StaN in very large-scale environments.

To assess the accuracy of the simulator, we first conducted a series of experiments with the same workload on both Splay and the simulator. For a given workload, we first created the overlays for each topic by having every node, either real or simulated, choose *viewSize* neighbors randomly. *viewSize* was configured such that the probability of the overlay being connected is 0.99 as specified in [33]. Finally, we ran StaN for eight rounds and collected results. We used rounds of 30 seconds for Splay, corresponding to a discrete time step of the simulator.

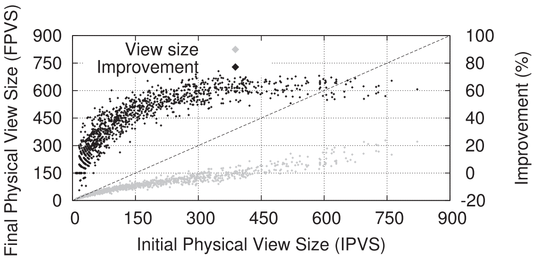
3.4 Performance

To assess StaN's performance in promoting link sharing, we defined two measurements: *logical view size* (LVS) and *physical view size* (PVS). LVS measures the number of logical links established by a node across all topics, which is the sum of the view sizes across all node's overlays. PVS captures the number of physical links that each node needs to establish. It is obtained by extracting the unique node identifiers from the logical views. Since StaN preserves the number of logical links, we expect LVS to remain constant and PVS to decrease as the protocol converges.

Fig. 6a presents the evolution of LVS and PVS for a synthetic universe with 100 nodes and 16 topics, for both the real and simulated environments. Values are averaged over five distinct runs. LVS remains constant across the whole experiment because StaN preserves the size of the views of individual overlays. PVS drops from 25 to around 15, which shows that StaN is able to share logical links after just a few rounds. The better performance of the simulator is explained by its discrete nature as the protocol runs in lock step mode, and nodes optimize their views before proceeding to the next round. Other reasons are the nonnegligible message loss and connectivity issues experienced in

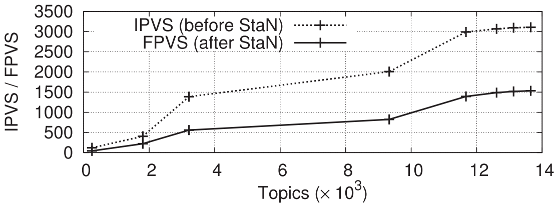


(a) $\mathcal{L}_8$ LiveJournal universe.

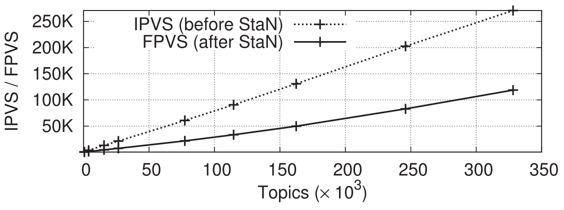


(b) $\mathcal{W}_8$ Wikipedia universe.

Fig. 7. IPVS/FPVS and relative improvement for the $\mathcal{L}_8$ and $\mathcal{W}_8$ universes.



(a) LiveJournal universes.



(b) Wikipedia universes.

Fig. 8. Evolution of the IPVS/FPVS with the number of topics.

PlanetLab due to faults and churn. Despite this slight deviation, one can expect that simulation results with larger universes will be representative of real-world deployments. Fig. 6b presents the cumulative distribution function of the view sizes of all nodes. Again, the results obtained by simulation mimic those of the real environment.

From this point on, we ran StaN in the simulated environment with the real universes to study its behavior in larger scales. All results below are the average of 10 independent runs. We consider both the *initial* PVS (IPVS) at the beginning of the experiment and the *final* PVS (FPVS) after running StaN.

The next experiment aims at observing the effectiveness of StaN at reducing PVS as a function of IPVS. This allows us to detect where the improvement happens, namely to which extent nodes with large view sizes benefit from StaN. Fig. 7 presents these results as a scatter plot: in the x-axis we have the IPVS and in the left y-axis the FPVS. The vast majority of points for the view size lie below the diagonal meaning that StaN effectively reduced node's view sizes. This is confirmed when we analyze the improvement in percentage (right y-axis). As expected, the improvement both in absolute and relative terms is greater for Wikipedia (bottom) due to its greater correlation. In fact, the majority of Wikipedia's nodes have an improvement over 40 percent, while for LiveJournal fewer nodes achieve such an improvement.

For some nodes, the improvement is negative. This only happens for nodes with very low IPVS and is because the weight function of StaN can sometimes split a physical link that was initially shared (by chance) when optimizing the overlay. This may happen for all nodes but it is only noticeable for nodes with very low IPVS. Indeed, the number of links that become physically shared on nodes with larger view sizes easily outweighs any alignment that may have occurred at creation time. Results for the other universes follow the same trend (not shown).

Finally, Fig. 8 presents a condensed view of the previous plots for LiveJournal universes $\mathcal{L}_1$-$\mathcal{L}_8$ and Wikipedia universes $\mathcal{W}_1$-$\mathcal{W}_8$. The results are obtained by extracting the IPVS and FPVS for each configuration, i.e., before and after running StaN. One can observe that the number of physical links grows much slower, by a factor of 2, with the number of topics when using StaN. This demonstrates that our approach is effective at scaling with the number of topics as it limits the number of physical links established by nodes with many subscriptions.

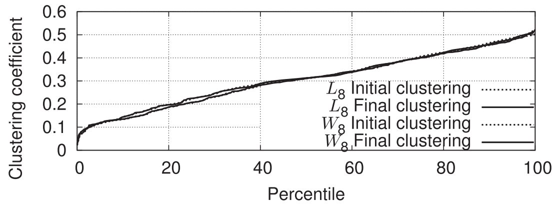
3.5 Fitness

We now study StaN's fitness by focusing on the structural properties of the overlays, namely *connectivity*, *clustering coefficient*, and *diameter* which affect reliability and effectiveness [11]. Therefore, StaN must not modify them with respect to the initial values of the OMPs. The experiments below are for the $\mathcal{L}_8$ and $\mathcal{W}_8$ universes, and results for the other configurations are similar (not shown). Values presented are the average of each property across all overlays.

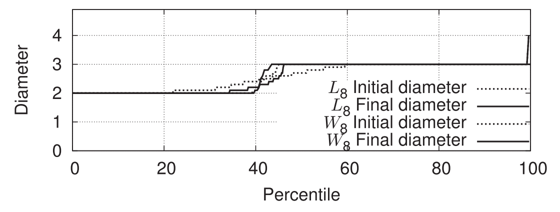
Connectivity. This property measures the number of connected components of each overlay. A single component indicates that the overlay is connected. By assumption, the initial overlays are managed by an OMP that creates a single connected component. In all the experiments conducted, we did not observe a single disconnection. This is due to the uniformity of the weight function, which ensures that every node is equally likely to be selected as a best neighbor, thus compensating the loss of links when nodes choose other neighbors.

Clustering coefficient. The results, presented in Fig. 9a, show the initial and final values as almost indistinguishable. In fact, due to the asymmetry of the weight function, the weight any two nodes assign to each other, or to a third, is completely unrelated, thus preventing nodes from selecting each other as neighbors, or preferring neighbors of neighbors.

Diameter. The results, presented in Fig. 9b, show that, as before, both distributions are similar. The reason lies in the way the weight function is designed. As neighbors are selected uniformly at random, the probability of losing links



(a) Clustering coefficient distribution.



(b) Diameter distribution.

Fig. 9. The clustering coefficient and diameter distribution for the $\mathcal{L}_8$ and $\mathcal{W}_8$ universes.

is identical to the probability of gaining links. Moreover, the asymmetry prevents the overlays from converging to a grid-like structure that would otherwise increase diameter when compared to the initial random overlay. Therefore, the randomness of link establishment provided by the weight function preserves the diameter of the overlays.

4 Related Work

Approaches to topic-based publish/subscribe can be generally divided in two categories: one that maintains multiple separate overlays per topic, and other that maintains a single general overlay.

In Scribe [34], each topic is managed by a single node in the distributed hash tables (DHT), the *rendezvous* point that handles subscription and unsubscription requests. Subscribers are organized in a multicast tree rooted at the *rendezvous* node that serves as the entry point to all events. CAN-multicast [35] also associates *rendezvous* node to topics. However, each topic is managed independently with a new protocol instance. Both approaches present scalability and fault-tolerance issues due to the existence of the *rendezvous* node. With both Scribe [34] and CAN-based multicast [35], nodes that are not interested in some topic may still act as forwarders, with they being internal nodes of the dissemination tree built on top of the DHT or the *rendezvous* node that implements the group membership management. Magnet [36] relies on the same principle but uses as substrate a DHT that clusters nodes according to interests. This design greatly reduces the load on forwarders, thus improving *accuracy*. StaN does not require the maintenance of a structured overlay network for its operation nor forwarder nodes. Magnet, similarly to Scribe of CAN-multicast, is more adapted to situations with a moderate number of well subscribed topics, while StaN and gossip-based approaches are more adapted to a large number of topics whose popularities follow a power-law distribution.

The daMulticast [27] departs from the structured approach to embrace a pure gossip-based strategy. Overlays are organized in a hierarchy, with an independent overlay per level, thus enabling completeness and accuracy. Probabilistic links are maintained from overlays at lower levels to their parents, thus reducing maintenance overhead and message complexity by exploiting the hierarchy of topics. This approach does, however, rely on these hierarchical relationships between topics. It would otherwise degrade to a traditional approach with a single overlay per topic. In contrast, StaN makes no assumptions on topic hierarchies but can nonetheless take advantage of them. As StaN works by exploiting individual nodes subscriptions instead of topic relationships, it is more flexible. TERA [8] relies on gossip-based protocols to maintain a general overlay, used for routing, and a separate overlay per topic, thus presenting scalability problems with high numbers of subscriptions.

To avoid the scalability problems of one overlay per topic, SpiderCast [5] uses a single overlay. Links are established according to two strategies: similarity among subscriptions or at random. To probabilistically ensure topic connectivity, the protocol attempts to guarantee that each node becomes k-covered for every topic it is interested in. Once a node becomes k-covered, SpiderCast does no longer search for nodes with closer interests. With full membership knowledge, this approach works well since chosen nodes are, by design, the most similar. When that is not the case, the performance degrades because the set of candidate nodes may not include the most similar ones. Still, SpiderCast is able to construct connected overlays with low degree knowing only 5 percent of the nodes. Moreover, the decision of link addition and removal must be made by two adjacent nodes due to the use of an undirected graph. The other main concern is service differentiation as it is not possible to offer different service levels based on topic requirements.

The *Min-TCO* problem [7] is defined as the construction of a graph with a minimum number of edges that ensures completeness and accuracy. Its decision version is shown to be NP-complete, and thus captures some inherent limitations of an approach based on a single overlay. While this results in overlays with low average view size (degree in the paper), the maximum view size can grow quite large. This is addressed by the low-TCO [37], which achieves both low average and maximum view sizes. Those approaches require however global knowledge, are computationally expensive, and do not support subscription dynamism. Recently, these issues have been tackled by divide-and-conquer strategies that enable parallelization dynamism [38], [39]. Still, they are centralized and designing a distributed equivalent is, to the best of our knowledge, an open issue.

A survey of proposals based on subscription correlation can be found in [40].

5 Concluding Remarks

In this paper, we presented StaN, a protocol that takes advantage of the correlation of interests among nodes in a topic-based publish-subscribe system with the goal of decreasing the number of physical links established. This is essential to promote topic scalability as the number of physical links established by a node represent an inherent limitation. StaN achieves this by the use of a clever weight function that allows overlays to retain their good structural properties, thus providing a robust and attractive infrastructure for data dissemination. In fact, by being oblivious to node subscriptions, StaN sidesteps the problems inherent to clustering but still offers considerable improvements in link sharing, thus emerging as a new point in the design space of topic-based publish-subscribe systems.

Our evaluation based on a real workload from traces of LiveJournal [16] and Wikipedia [17] with simulations and a real deployment on PlanetLab shows that StaN achieves its performance and fitness goals. Performance is ultimately limited by the initial view size of the nodes and topic correlation. As expected, the benefits of StaN are more evident in nodes with large views, precisely those with scalability problems, as it effectively reduces the number of physical links maintained

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their valuable comments and constructive suggestions to improve the quality of the paper.

REFERENCES

[1] P. Eugster, P. Felber, R. Guerraoui, and A.-M. Kermarrec, "The Many Faces of Publish/Subscribe," *ACM Computing Survey*, vol. 35, no. 2, pp. 114-131, 2003.
[2] A. Nunes, J. Marques, and J. Pereira, "Seeds: The Social Internet Feed Caching and Dissemination Architecture," *Proc. INForum Simpósio de Informática*, 2009.
[3] S. Jun and M. Ahamad, "Feedex: Collaborative Exchange of News Feeds," *Proc. Int'l Conf. World Wide Web*, 2006.
[4] J. Patel, E. Rivière, I. Gupta, and A.-M. Kermarrec, "Rappel: Exploiting Interest and Network Locality to Improve Fairness in Publish-Subscribe Systems," *Computer Networks*, vol. 53, no. 13, pp. 2304-2320, Aug. 2009.
[5] G. Chockler, R. Melamed, Y. Tock, and R. Vitenberg, "Spidercast: A Scalable Interest-Aware Overlay for Topic-Based Pub/Sub Communication," *Proc. Int'l Conf. Distributed Event-Based Systems*, 2007.
[6] S. Voulgaris, E. Rivière, A.-M. Kermarrec, and M. van Steen, "Sub-2-Sub: Self-Organizing Content-Based Publish Subscribe for Dynamic Large Scale Collaborative Networks," *Proc. Int'l Workshop Peer-to-Peer Systems*, 2006.
[7] G. Chockler, R. Melamed, Y. Tock, and R. Vitenberg, "Constructing Scalable Overlays for Pub-Sub with Many Topics," *Proc. 26th Ann. ACM Symp. Principles of Distributed Computing*, 2007.
[8] R. Baldoni, R. Beraldi, V. Quema, L. Querzoni, and S. Tucci-Piergiovanni, "TERA: Topic-Based Event Routing for Peer-to-Peer Architectures," *Proc. Int'l Conf. Distributed Event-Based Systems*, 2007.
[9] K. Birman, M. Hayden, O. Ozkasap, Z. Xiao, M. Budiu, M. Mihai, and Y. Minsky, "Bimodal Multicast," *ACM Trans. Computer Systems*, vol. 17, no. 2, pp. 41-88, 1999.
[10] P. Eugster, R. Guerraoui, S. Handurukande, P. Kouznetsov, and A.-M. Kermarrec, "Lightweight Probabilistic Broadcast," *ACM Trans. Computer Systems*, vol. 21, no. 4, pp. 341-374, 2003.
[11] M. Jelasity, S. Voulgaris, R. Guerraoui, A.-M. Kermarrec, and M. van Steen, "Gossip-Based Peer Sampling," *ACM Trans. Computer Systems*, vol. 25, no. 3, article 8, Aug. 2007.
[12] R. Chand and P. Felber, "Semantic Peer-to-Peer Overlays for Publish/Subscribe Networks," *Proc. Int'l Conf. Parallel and Distributed Computing*, 2005.
[13] M. Jelasity, A. Montresor, and O. Babaoglu, "T-Man: Gossip-Based Fast Overlay Topology Construction," *Int'l J. Computer and Telecomm. Networking* , vol. 53, no. 13, pp. 2321-2339, Aug. 2009.
[14] L. Massoulié, A.-M. Kermarrec, and A. Ganesh, "Network Awareness and Failure Resilience in Self-Organising Overlay Networks," *Proc. Symp. Reliable Distributed Systems*, 2003.
[15] "Planetlab," http://www.planet-lab.org, 2013.
[16] http://www.livejournal.com/stats.bml, 2013.
[17] "Wikipedia Database Dumps," http://dumps.wikimedia.org/, 2013.
[18] M. Matos, A. Nunes, R. Oliveira, and J. Pereira, "Stan: Exploiting Shared Interests without Disclosing Them in Gossip-Based Publish/Subscribe," *Proc. Ninth Int'l Workshop Peer-to-Peer Systems (IPTPS '10)*, 2010.
[19] M. Matos, P. Felber, R. Oliveira, J. Pereira, and E. Rivière, "Scaling Up Publish/Subscribe Overlays Using Interest Correlation for Link Sharing (Supplemental Document)," *(TO BE COMPLETED FOR THE CAMERA READY)*.
[20] H. Liu, V. Ramasubramanian, and E. Sirer, "Client Behavior and Feed Characteristics of RSS, a Publish-Subscribe System for Web Micronews," *Proc. Internet Measurement Conf.*, 2005.
[21] P. Fraigniaud, P. Gauron, and M. Latapy, "Combining the Use of Clustering and Scale-Free Nature of Exchanges into a Simple and Efficient P2P System," *Proc. Int'l Conf. Parallel and Distributed Computing*, 2005.
[22] S. Saroiu, P.K. Gummadi, and S.D. Gribble, "A Measurement Study of Peer-to-Peer File Sharing Systems," *Proc. ACM/SPIE Multimedia Computing and Networking*, 2002.
[23] S. Handurukande, A.-M. Kermarrec, F. Le Fessant, L. Massoulié, and S. Patarin, "Peer Sharing Behaviour in the eDonkey Network, and Implications for the Design of Server-Less File Sharing Systems," *Proc. First ACM SIGOPS/EuroSys European Conf. Computer Systems*, 2006.
[24] A.J. Ganesh, A.-M. Kermarrec, and L. Massoulié, "SCAMP: Peer-to-Peer Lightweight Membership Service for Large-Scale Group Communication," *Proc. Third Int'l COST264 Workshop Networked Group Comm. (NGC '01)*, pp. 44-55, 2001.
[25] P. Eugster, R. Guerraoui, A.-M. Kermarrec, and L. Massoulié, "From Epidemics to Distributed Computing," *Computer*, vol. 37, no. 5, pp. 60-67, May 2004.
[26] J. Pereira, L. Rodrigues, R. Oliveira, and A.-M. Kermarrec, "NeEM: Network-Friendly Epidemic Multicast," *Proc. Symp. Reliable Distributed Systems*, 2003.
[27] S. Baehni, P. Eugster, and R. Guerraoui, "Data-Aware Multicast," *Proc. Int'l Conf. Dependable Systems and Networks*, 2004.
[28] M. Luby, *Pseudorandomness and Cryptographic Applications*. Princeton Univ. Press, 1994.
[29] C. Gkantsidis, M. Mihail, and A. Saberi, "Random Walks in Peer-to-Peer Networks: Algorithms and Evaluation," *Performance Evaluation - P2P Computing Systems*, vol. 63, no. 3, pp. 241-263, 2006.
[30] D. Karger, E. Lehman, T. Leighton, R. Panigrahy, M. Levine, and D. Lewin, "Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web," *Proc. 29th Ann. ACM Symp. Theory Computing (STOC '97)*, 1997.
[31] L. Adamic and B. Huberman, "Zipf's Law and the Internet," *Glottometrics*, vol. 3, no. 1, pp. 143-150, 2002.
[32] L. Leonini, E. Rivière, and P. Felber, "SPLAY: Distributed Systems Evaluation Made Simple (or How to Turn Ideas into Live Systems in a Breeze)," *Proc. Symp. Networked Systems Design and Implementation*, 2009.
[33] A.-M. Kermarrec, L. Massoulié, and A. Ganesh, "Probabilistic Reliable Dissemination in Large-Scale Systems," *IEEE Trans. Parallel and Distributed Systems*, vol. 14, no. 3, pp. 248-258, Mar. 2003.
[34] M. Castro, P. Druschel, A. Kermarrec, and A. Rowstron, "SCRIBE: A Large-Scale and Decentralized Application-Level Multicast Infrastructure," *IEEE J. Selected Areas Comm.*, vol. 20, no. 8, pp. 1489-1499, Oct. 2002.
[35] S. Ratnasamy, M. Handley, R.M. Karp, and S. Shenker, "Application-Level Multicast Using Content-Addressable Networks," *Proc. Workshop Networked Group Comm.*, 2001.
[36] S. Girdzijauskas, G. Chockler, Y. Vigfusson, Y. Tock, and R. Melamed, "Magnet: Practical Subscription Clustering for Internet-Scale Publish/Subscribe," *Proc. Int'l Conf. Distributed Event-Based Systems*, 2010.
[37] M. Onus and A.W. Richa, "Parameterized Maximum and Average Degree Approximation in Topic-Based Publish-Subscribe Overlay Network Design," *Proc. IEEE Int'l Conf. Distributed Computing Systems*, 2010.
[38] C. Chen, H.-A. Jacobsen, and R. Vitenberg, "Divide and Conquer Algorithms for Publish/Subscribe Overlay Design," *Proc. IEEE Int'l Conf. Distributed Computing Systems*, 2010.

[39] C. Chen, R. Vitenberg, and H.-A. Jacobsen, "Scaling Construction of Low Fan-out Overlays for Topic-Based Publish/Subscribe Systems," *Proc. IEEE Int'l Conf. Distributed Computing Systems,* 2011.

[40] L. Querzoni, "Interest Clustering Techniques for Efficient Event Routing in Large-Scale Settings," *Proc. Int'l Conf. Distributed Event-Based Systems,* 2008.

Miguel Matos received the MSc degree in computer science from the University of Minho in July 2009 with the thesis network-aware epidemic broadcast. He is working toward the PhD degree at the University of Minho, Portugal. His research interests include the problems that arise in very large-scale distributed systems, namely on the development and application of peer-to-peer and epidemic-based techniques to data dissemination and management. He is a student member of the IEEE and the IEEE Computer Society and a member of the ACM.

Pascal Felber received the MSc and PhD degrees in computer science from the Swiss Federal Institute of Technology (EPFL). He was at Oracle Corporation and Bell-Labs in US, and at Institut EURECOM in France. Since 2004, he has been a professor of computer science at the University of Neuchâtel, Switzerland, working in the field of dependable, distributed, and concurrent systems. He has published more than 80 research papers in various journals and conferences. He is a member of the IEEE.

Rui Oliveira received the PhD degree in computer science from the Swiss Federal Institute of Technology (EPFL) in 2000. He is an associate professor of computer science at Universidade do Minho. His main research contributions have been in the fields of fault-tolerant distributed agreement and epidemic multicast algorithms and in the conception, development and assessment of replicated database systems. He has coordinated several projects on scalable data management and dissemination. He is a member of the IEEE.

José O. Pereira received the graduate degree in computer engineering and informatics in 1995. He received the master's degree and the PhD degree in computer science from the University of Minho in 1998 and 2002, respectively. He is currently an assistant professor at the Computer Science Department, University of Minho. He has been doing research in reliable distributed systems, in particular, in group communication and its application to database replication.

Etienne Rivière received the PhD degree in computer science from the University of Rennes, France, in November 2007. In 2008 and 2009, he has been a postdoctoral fellow under an "Alain Bensoussan" grant from ERCIM, which led him to the University of Neuchâtel and to NTNU Trondheim in Norway. He is a lecturer at the University of Neuchâtel, Switzerland. His research interests include the design, analysis, implementation and evaluation of large-scale distributed systems, and concurrent systems. He is a member of the IEEE, ACM, and Usenix.