Exploiting the Synergy Between Gossiping and Structured Overlays

Ali Ghodsi
Swedish Institute of Computer Science (SICS)
Box 1263
SE–16429 Kista
Sweden
ali(at)sics.se

Seif Haridi
KTH—Royal Institute of Technology
Electrum 229
SE–16440 Kista
Sweden
haridi(at)kth.se

Hakim Weatherspoon
Cornell University
Computer Science Department
Ithaca, NY 14853
USA
hweather(at)cs.cornell.edu

ABSTRACT

In this position paper we argue for exploiting the synergy between gossip-based algorithms and structured overlay networks (SON). These two strands of research have both aimed at building fault-tolerant, dynamic, self-managing, and large-scale distributed systems. Despite the common goals, the two areas have, however, been relatively isolated. We focus on three problem domains where there is an untapped potential of using gossiping combined with SONs. We argue for applying gossip-based membership for ring-based SONs—such as Chord and Bamboo—to make them handle partition mergers and loopy networks. We argue that small world SONs—such as Accordion and Mercury—are specifically well-suited for gossip-based membership management. The benefits would be better graph-theoretic properties. Finally, we argue that gossip-based algorithms could use the overlay constructed by SONs. For example, many unreliable broadcast algorithms for SONs could be augmented with anti-entropy protocols. Similarly, gossip-based aggregation could be used in SONs for network size estimation and load-balancing purposes.

Keywords

Gossip-based Algorithms, Structured Overlay Networks, Distributed Hash Tables

1. INTRODUCTION

Due to the scale and dynamicity of many applications running over the Internet, it has become increasingly important to build systems that are scalable, fault-tolerant, and self-managing. Two strands of research have addressed these issues: *gossip-based algorithms* and *structured overlay networks* (SONs). Gossip-based algorithms have proved to be a powerful way to achieve the above requirements, as well as being effective in solving many problems, such as broadcast [5], failure detection [40], and aggregation [21]. Furthermore, they are simple, which makes them easy to implement and maintain. At the same time, research on *structured overlay networks* (SONs) have flourished and a plethora of such systems provide scalability, fault-tolerance, and self-management. Most SONs boil down to providing a *distributed hash table* (DHT) abstraction, or group communication capability.

In this paper we take the position that there is an untapped potential in the synergy between gossip-based algorithms and SONs. We believe that these two research areas, despite their similarities and common goals, have been isolated from each other. Most research papers in either of the areas, do not reference papers in the other area. There are a few exceptions to this, which we will highlight. We point to open problems that can be best addressed if gossip-based algorithms are combined with SONs. We also point at existing work, which we believe can be improved by the cross-fertilization of these areas.

1.1 Outline

Next, Section 2 briefly overviews gossip-based algorithms and SONs. Thereafter, Section 3 describes three problem domains, which we think would benefit from the outlined research. Section 4 describes existing related work that already combines gossip-based algorithms with SONs, and their respective advantages. Finally, Section 5 concludes.

2. BACKGROUND

In this section we briefly give an overview gossip-based algorithms and structured overlay networks.

2.1 Gossip Algorithms

An algorithm is gossip-based if it prescribes nodes to periodically pick a neighbor, and exchange information with that node. Moreover, the amount of data exchanged has a bounded size. Typically, the information exchanged quickly disseminates to all nodes through a process that resembles gossiping or epidemics [9, 12].

It is often assumed that nodes are picked at *random* during the gossip process. This is achieved by using a *membership* service, where each node maintains routing pointers to other nodes, such that every node can pick another random node with uniform probability [41, 20]. This service can itself be implemented using a gossip-based algorithm that periodically makes nodes exchange routing pointers with each other.

If there is no requirement of randomness in the picking of nodes, many traditional algorithms are gossip-based. In particular, most *self-stabilizing* [10] algorithms fit the definition of gossip-based algorithms.

Gossip-based algorithms have been used to solve many problems. For example, Birman *et al.* [5] use a gossip-based protocol to implement a probabilistic reliable multicast service, whose throughput—unlike traditional algorithms—is not as affected by failures. Van Renesse *et al.* [40] use a gossip-based protocol to spread information about heartbeats to implement a scalable failure detector. Others, such as Jelasity *et al.* [22], use gossip-protocols to aggregate information—such as the average, maximum, or minimum—at all nodes.

2.2 Structured Overlay Networks

All structured overlay networks make use of the concept of an *identifier space* consisting of a set of *identifiers*, which we take to be positive integers less than some large constant value. At all times, each identifier is under the *responsibility* of one node[1]. Each node maintains routing pointers to other nodes such that every node can find the node responsible for a given identifier. The process of finding responsible nodes is referred to as the *lookup operation*. It is typically guaranteed that the number of routing pointers, as well as the number of redirects a lookup needs, is $O(\log n)$ for n nodes, even though other schemes are possible. This is typically implemented by assigning to each node an identifier, and connecting nodes to each other to form a distributed *ring structure*[2]. Lookups essentially traverse the ring structure to find responsible nodes. To avoid the worst case of traversing the whole ring, each node can maintain extra pointers across the ring structure, which can be used to speed up the lookup operation. For example in Chord the extra pointers at a node p are placed with exponentially increasing distances from p in the identifier space. Hence, the distance to any destination can always be at least halved in one hop, yielding a worst-case of $O(\log n)$ hops to reach any destination.

SONs are mainly used for implementing two services: *distributed hash tables* (DHTs) and *group communication*. A DHT is a hash table abstraction built on top of the SON. The DHT is implemented by hashing the key of each item (key/value pair) such that it gets an identifier from the identifier space. Every item is stored at the node responsible for the item's identifier. Hence, lookups can be used to resolve keys.

SONs can also be used for group communication [14, 11, 15, 3]. For example, uninformed search can be done by broadcasting on top of the routing pointers of a SON. Similarly, overlay multicast can be implemented by creating a SON instance for every multicast group. Nodes interested in a multicast group join that group, and nodes wanting to multicast a message simply broadcast the message within the SON representing the desired multicast group. Broadcast on top of SONs exploits the structure of the overlay to avoid transmitting redundant messages. Hence, they not only guarantee non-redundant delivery of messages, but also non-redundant message reception.

One of the main uses of DHTs have been name-resolution. In particular, it is useful to be able to find out the current address or location of a user or node. In P-Grid, the DHT is used to find out the current IP address of a node that has left the system and later returned with a new IP address [2]. Similar uses of DHTs can be found for the Host Identity Protocol (HIP) [32], Session Initiation Protocol (SIP) [31], and I^3 [38]. It is evident that name resolution, for dealing with mobility and dynamic IP addresses, is relevant in purely gossip-based systems too. Hence, using DHTs in such systems would be advantageous.

3. GOSSIP ALGORITHMS FOR IMPROVING STRUCTURED OVERLAYS

Gossip algorithms have had an important role in the building of robust SONs, even though it is not widely acknowledged. Rhea *et al.* [35] made some of the first real experiments with existing SON implementations, and found that the implementations of the two main SONs—Chord and Pastry—did not work properly under churn. They devised a new system called Bamboo, which provides a new algorithm for maintaining the ring structure. Even though not explicitly mentioned by the authors, the algorithm is gossip-based. The algorithm maintains the ring structure by having each node keeping a *leaf set*, which contains the closest nodes on the ring. The nodes periodically gossip information about the contents of the leaf-set with each other. Early versions of Bamboo used a similar algorithm that was not periodic. Nodes reactively sent notifications to the nodes in their leaf sets about changes in their neighborhood. It therefore suffered from the network becoming overly congested due to positive feedback cycles created by the sending of the notifications. Hence, the use of a gossip protocol avoided such bursty behavior because of its periodic behavior and bounded message exchange. These reasons have previously been the main motivation for using gossip algorithms in other contexts [5].

The seminal work by Stoica *et al.* on Chord [39] describes an algorithm called *periodic stabilization* that maintains a ring structure. Periodic stabilization resembles a gossip algorithm, as each node periodically gossips with its immediate successor on the ring about information about its neighborhood. Though simple, the algorithm ensures that any number of concurrent joins will eventually lead to a perfect ring. It is also very robust to node failures.

Periodic stabilization has, however, two shortcomings. First, it cannot properly heal from network partitions. When network partitions happen, periodic stabilization ensures that each partitioned component eventually forms a perfect ring. When the network partition ceases, periodic stabilization cannot efficiently merge the disjoint rings.

The second shortcoming of periodic stabilization is due to failure scenarios which might cause the ring to enter a state, referred to as *loopy*, in which a perfect ring is never formed

[1]Some systems, such as P-Grid [1], have multiple nodes responsible for the same identifier. The same applies to symmetric replication [16].

[2]The ring structure is a distributed doubly-linked list with its header and tail connected.

again. In a loopy state, every node p points to another node q, which it believes is its successor, and node q believes p to be its predecessor. But to traverse all nodes in the ring, the successor pointers have to be traversed more than once around the whole ring (see Figure 1). Determining whether the ring is in a loopy state is a global property. Hence, there has been two solutions up to now, both requiring sending messages to all nodes in the system [27, 24].

3.1 Healing From Merged Partitions and Loopy Networks

Gossip-based protocols have previously been successfully used to handle network partitions. We believe the reasons for this to be threefold. First, its periodic behavior is ideal for detecting sudden state changes, such as the merger of a network partition or the detection of a loopy network. Second, its bounded resource utilization ensures that sudden global changes, such as that of a merger of network partitions, avoids positive feedback cycles in very much the way explained in the previous Bamboo scenario. Third, the exponential speed at which information is disseminated ensures efficient topology recovery. Our position is that gossip-based protocols can be used to tackle both mentioned problems in SONs.

There are already several gossip-based algorithms that are viable for being used to tackle the aforementioned problems with the ring topology of SONs. In particular, T-man [19, 30] can build various desired topologies from a completely random graph. T-chord [30] can build the overlay of Chord from a random graph. Interestingly, these algorithms are efficient as the desired network topology appears after $O(\log n)$ rounds of gossiping, while doing periodic bounded message exchange. A SON could use a gossip-based algorithm, such as T-chord [30], and periodically check if T-chord has better information that can be incorporated into the structured overlay. The T-chord protocol would then run in epochs, which are restarted periodically as prescribed by Jelasity *et al.* [21, 19]. Shafaat *et al.* [37] have recently showed that the problem of merging SONs after partitions can be solved using gossip-based algorithms, there is, however, room for improvement.

3.2 Membership for SONs

The SON requirement of maintaining a ring structure is quite strict. Some of the initial SONs, such as Chord [39] and CAN [34], had also a strict requirement on which nodes the extra pointers should be pointing to. Since the design allowed for little flexibility in where pointers could be placed, it made little sense to use gossip-based algorithms for membership management[3]. We believe that this has gradually changed, which makes the usage of gossip-based algorithms for membership management in SONs viable. Already the Pastry system [36], had flexibility in how nodes could be interconnected. Later it was also shown how systems, such as Chord, could be changed to allow for flexibility in how the nodes are interconnected. In particular Gummadi *et al.* [17] described that it was sufficient for each node to partition the identifier space into intervals of exponentially increasing sizes, and have one pointer anywhere in each interval.

[3] A membership protocol maintains the routing pointers of a node

Several recent SONs only prescribe that nodes should be connected to each other according to a probability distribution that ensures a small-world property. This applies to SONs such as Accordion [26], Mercury [4], Symphony [29], and EpiChord [25]. These systems ensure that the diameter of the overlay is low, by having each node pick a neighbor with a probability inversely proportional to the distance to the neighbor, as described by Kleinberg [23]. The graph theoretic properties of these systems, has not been as thoroughly investigated as those of gossip-based membership protocols.

Membership protocols have extensively been studied in the context of gossip-based algorithms. They have been used to create different types of topologies, such as random [41, 13], clustered [42, 33], or application specific [19]. For example, Cyclon [41] ensures that each node has routing pointers to a small subset of the nodes in the system, such that the nodes in the subset are picked with uniform distribution across all nodes in the system. Furthermore, the graph induced by the membership protocol exhibits other attractive properties, such as robustness, low clustering coefficient, and highly symmetric and similar in and out-degrees. The graph theoretic properties of these systems have been studied [20], and undesirable properties remedied.

We believe that the membership protocols of flexible SONs, such as the ones mentioned above, have not undergone proper investigation. Most of them perform membership management heuristically, by, for example, discovering membership information through lookups performed by the application. Our position is that the flexibility of such SONs make them ideal for gossip-based membership protocols. This requires that the membership protocols can ensure that nodes are picked according to the probability distribution prescribed by the SON. We believe it is viable that gossip-based protocols can achieve this, since gossip-based membership already has been used to construct many different types of graphs. For example, T-man [19] can be used with a ranking function that ranks nodes according to the probability distribution described by the small-world SON (see T-man [19] for more details). Since many of the existing gossip-based based membership protocols have exhibited desirable properties, these could perhaps be transferred to SONs, if their topology is maintained by a gossip-based membership protocol.

3.3 Gossiping on top of a SON

Researchers from the SON community have suggested to run gossip-based algorithms on top of SONs. For example, instead of using a gossip-based membership service that creates a random graph, a SON like Accordion [26] can be used. Even if there might be no intrinsic value in using SONs for gossip-based algorithms, it might be that a SON is already chosen to solve a given problem and one would like to use gossip-based algorithms on top of the existing SON. One can then also utilize the SON for gossiping.

Any gossip-based algorithm can be deployed on top of a SON. The gossip-based algorithm can use the routing pointers of the SON for random gossip exchange. We mention two approaches that can be used in cases where it is required that nodes are picked with uniform probability for

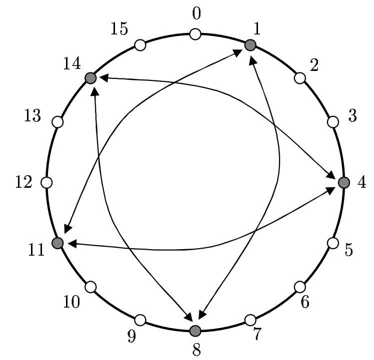

Figure 1: A loopy ring from which periodic stabilization cannot recover. The dark circles represent nodes in the ring, and each doubly arrowed link represents a successor pointer and a predecessor pointer.

gossip exchange. First, a random identifier can be picked, and a lookup can be made to find the node responsible for that identifier. Second, a random walk can be made to find random peers.

In Structella [6], Gnutella is built on top of the SON Pastry [36], and flooding and random walks are used for searching. Castro *et al.* [7] show how this approach can lead to lower membership maintenance cost. By exploiting the structure of the SON, the flooding algorithms used for searching can completely avoid the sending of redundant messages.

The work on flooding on top of SONs could be complemented with gossip-based algorithms. For example, the broadcast algorithms for SONs [14, 11] are costly to make reliable. Instead, one could use the unreliable broadcast algorithm for SONs, and then deploy a gossip-based *anti-entropy* [9, 5] protocol in the background to achieve reliability. The anti-entropy protocol lets nodes gossip among each other to find out whether they are missing any information, which then can be transmitted. Hence, the broadcast algorithm, which guarantees to never send redundant messages, will attempt to disseminate information to all nodes, while anti-entropy protocol ensures that all nodes with high probability and low cost get all the messages.

Finally, we think that the use of aggregates in SONs would benefit from using gossip-based protocols. For example, most SONs require nodes to have an estimate of the number of nodes, n, present in the system. This can be used to maintain routing tables, whose size are dependent on n. Most SONs have very rudimentary algorithms for estimating the network size. For example, in Viceroy [28] and Mercury [4], each node's distance to its successor in the identifier space is used to calculate the number of nodes in the system. There is, however, no analysis of the accuracy of such an estimate. In contrast, Jelasity *et al.* [22] show how their aggregation algorithm for calculating averages can be used to exactly calculate the network size within $O(\log n)$ rounds. Similarly, gossip-based aggregation could be used to calculate the average, maximum, or minimum load on nodes. This information could effectively be used for load-balancing.

We give an example of how gossip-based algorithms can exploit SON properties. The following algorithm estimates the network size in a SON by using the gossip-based aggregation algorithm described by Jelasity *et al.* [22]. Each node i starts with an initial value v_i, which is set to the distance to its successor on the ring. Then a gossip algorithm is applied to get the average inter-node distance, δ, on the identifier space. The network size is then the ratio between the identifier space size and δ. The system should at all times maintain the invariant that the sum of all the estimates v_i is equal to the size of the identifier space. Hence, a joining node sets its estimate to zero. Similarly, a leaving node i ensures that some other j sets $v_j = v_j + v_i$. Failures are dealt with by restarting the algorithm, similarly as described by Jelasity *et al.* [22].

4. RELATED GOSSIP-BASED SONS

Two existing SONs—P-Grid [1] and Kelips [18]—are completely gossip-based, though this fact is not widely known in the two communities.

P-Grid [1], which is one of the first SONs, is completely gossip-based. It is indeed a SON as at each instant in time, it is possible to make a lookup to find the node responsible for any given key. At the same time, data is spread to replicas using a gossip-based protocol. Decisions, such as when the replication degree of an item should be increased, are triggered by gossip-based protocol. As a direct result of this, P-Grid can heal from merged network partitions [8]. Moreover, it load-balances as nodes storing popular items are automatically replicated by other nodes.

Kelips [18] is another gossip-based SON. Each node in Kelips picks a random group number between $[1, \sqrt{n}]$, where n is the network size. A gossip protocol is used to ensure that each node has $\sqrt{n}$ routing pointers to all nodes with the same group number, and one routing pointer to at least one node from every other group number. Items are hashed to receive a group number, and an item with group number i is stored at all nodes with group number i. This way, a lookup can in one step find the node responsible for any provided identifier. Replication of data within a group is done using a gossip-based protocol. The main cost of Kelips is thus in maintaining $O(\sqrt{n})$ replicas of each item, where n is the network size. This could cause problems when there are many updates in the system. Nevertheless, Kelips, in similarity with P-Grid, has the advantage of being resilient to network partitions and automatically does load-balancing.

5. CONCLUSION

We have argued for the synergy of gossip-based algorithms and structured overlay networks (SON). Both research efforts have had similar goals: building fault-tolerant, dynamic, self-managing, and large-scale distributed systems. Despite their common goals, the two areas have, however, been relatively isolated. We explicitly solicited research on three common connecting points: gossip-based ring maintenance in SONs, membership in small-world SONs, and gossiping on top of SONs. We argued that applying gossip-based membership for ring-based SONs—such as Chord and Bamboo—will make them self-heal from partition mergers and loopy networks. We argued that small world SONs—such as Accordion and Mercury—would have better graph-theoretic properties if they used gossip-based membership management. Last, we mentioned several ways in which gossip-based algorithms could beneficially run on top of the overlay of a SON. For example, many unreliable broadcast algorithms for SONs could be augmented with anti-entropy protocols to achieve reliability. Furthermore, many SONs could benefit from using gossip-based aggregation protocols for network size estimation and load-balancing.

Acknowledgments

We would like to thank Jim Dowling and Sverker Janson at SICS for their comments on earlier drafts of this paper. This work has been financed by VINNOVA 2005-02512 Trust-Dis, SICS Center for Networked Systems (CNS), and IST project SELF-MAN.

6. REFERENCES

[1] K. Aberer, P. Cudré-Mauroux, A. Datta, Z. Despotovic, M. Hauswirth, M. Punceva, and R. Schmidt. P-Grid: a self-organizing structured P2P system. *SIGMOD Record*, 32(3):29–33, 2003.

[2] K. Aberer, A. Datta, and M. Hauswirth. Efficient, self-contained handling of identity in Peer-to-Peer systems. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 16(7):858–869, 2004.

[3] L. O. Alima, A. Ghodsi, P. Brand, and S. Haridi. Multicast in DKS(N, k, f) Overlay Networks. In *The 7th International Conference on Principles of Distributed Systems (OPODIS'03)*, volume 3144 of *Lecture Notes in Computer Science (LNCS)*, pages 83–95. Springer-Verlag, 2004.

[4] A. R. Bharambe, M. Agrawal, and S. Seshan. Mercury: Supporting Scalable Multi-Attribute Range Queries. In *Proceedings of the ACM SIGCOMM 2004 Symposium on Communication, Architecture, and Protocols*, pages 353–366, Portland, OR, USA, March 2004. ACM Press.

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

[6] M. Castro, M. Costa, and A. Rowstron. Should we build Gnutella on a structured overlay? *SIGCOMM Computing Communication Review*, 34(1):131–136, 2004.

[7] M. Castro, M. Costa, and A. Rowstron. Debunking Some Myths About Structured and Unstructured Overlays. In *Proceedings of the 2nd USENIX Symposium on Networked Systems Design and Implementation (NSDI'05)*, Boston, MA, USA, May 2005. USENIX.

[8] A. Datta and K. Aberer. The Challenges of Merging Two Similar Structured Overlays: A Tale of Two Networks. In *Proceedings of the First International Workshop on Self-Organizing Systems (IWSOS'06)*, volume 4124 of *Lecture Notes in Computer Science (LNCS)*, pages 7–22. Springer-Verlag, 2006.

[9] A. Demers, D. Greene, C. Hauser, W. Irish, J. Larson, S. Shenker, H. Sturgis, D. Swinehart, and D. Terry. Epidemic Algorithms for Replicated Database Maintenance. In *Proceedings of the 7th Annual ACM Symposium on Principles of Distributed Computing (PODC'87)*, pages 1–12, New York, NY, USA, 1987. ACM Press.

[10] E. W. Dijkstra. Self Stabilization in spite of Distributed Control. *Communications of the ACM*, 17(11):643–644, 1974.

[11] S. El-Ansary, L. O. Alima, P. Brand, and S. Haridi. Efficient Broadcast in Structured P2P Netwoks. In *Proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS'03)*, volume 2735 of *Lecture Notes in Computer Science (LNCS)*, pages 304–314, Berkeley, CA, USA, 2003. Springer-Verlag.

[12] P. T. Eugster, R. Guerraoui, A.-M. Kermarrec, and L. Massoulie. From epidemics to distributed computing. *IEEE Computer*, 37:60–67, May 2004.

[13] A. Ganesh, A. Kermarrec, and L. Massoulie. Peer-to-Peer Membership Management for Gossip-based Protocols. *ACM Transactions on Computer Systems (TOCS)*, 52(2):139–149, 2003.

[14] A. Ghodsi. *Distributed k-ary System: Algorithms for Distributed Hash Tables*. PhD dissertation, KTH—Royal Institute of Technology, Stockholm, Sweden, December 2006.

[15] A. Ghodsi, L. O. Alima, S. El-Ansary, P. Brand, and S. Haridi. Self-Correcting Broadcast in Distributed Hash Tables. In *Proceedings of the 15th International Conference, Parallel and Distributed Computing and Systems*, Marina del Rey, CA, USA, November 2003.

[16] A. Ghodsi, L. O. Alima, and S. Haridi. Symmetric Replication for Structured Peer-to-Peer Systems . In *Proceedings of the 3rd International VLDB Workshop on Databases, Information Systems and Peer-to-Peer Computing (DBISP2P'05)*, volume 4125 of *Lecture Notes in Computer Science (LNCS)*, pages 74–85. Springer-Verlag, 2005.

[17] K. Gummadi, R. Gummadi, S. Gribble, S. Ratnasamy, S. Shenker, and I. Stoica. The impact of DHT routing geometry on resilience and proximity. In *Proceedings of the ACM SIGCOMM 2003 Symposium on Communication, Architecture, and Protocols*, pages 381–394, New York, NY, USA, 2003. ACM Press.

[18] I. Gupta, K. Birman, P. Linga, A. Demers, and R. van Renesse. Kelips: Building an Efficient and Stable P2P DHT Through Increased Memory and Background

Overhead. In *Proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS'03)*, volume 2735 of *Lecture Notes in Computer Science (LNCS)*, pages 160–169, Berkeley, CA, USA, 2003. Springer-Verlag.

[19] M. Jelasity and Ö. Babaoglu. T-man: Gossip-based overlay topology management. In *Proceedings of 3rd Workshop on Engineering Self-Organising Systems (EOSA'05)*, volume 3910 of *Lecture Notes in Computer Science (LNCS)*, pages 1–15. Springer-Verlag, 2005.

[20] M. Jelasity, R. Guerraoui, A.-M. Kermarrec, and M. van Steen. The peer sampling service: experimental evaluation of unstructured gossip-based implementations. In *Proceedings of the 5th ACM/IFIP/USENIX International Conference on Middleware (MIDDLEWARE'04)*, volume 3231 of *Lecture Notes in Computer Science (LNCS)*, pages 79–98, New York, NY, USA, 2004. Springer-Verlag.

[21] M. Jelasity and A. Montresor. Epidemic-Style Proactive Aggregation in Large Overlay Networks. In *Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'04)*, pages 102–109, Tokyo, Japan, March 2004. IEEE Computer Society.

[22] M. Jelasity, A. Montresor, and Ö. Babaoglu. Gossip-based Aggregation in Large Dynamic Networks. *ACM Transactions on Computer Systems (TOCS)*, 23(3), August 2005.

[23] J. M. Kleinberg. The small-world phenomenon: an algorithm perspective. In *Proceedings of the 32nd ACM Symposium on Theory of Computing (STOC'00)*, pages 163–170, Portland, OR, USA, 2000. ACM Press.

[24] B. Leong and J. Li. Achieving One-Hop DHT Lookup and Strong Stabilization by Passing Tokens. In *12th International Conference on Networks (ICON'04)*, Singapore, November 2004. IEEE Computer Society.

[25] B. Leong, B. Liskov, and E. Demaine. EpiChord: Parallelizing the Chord Lookup Algorithm with Reactive Routing State Management. In *12th International Conference on Networks (ICON'04)*, Singapore, November 2004. IEEE Computer Society.

[26] J. Li, J. Stribling, R. Morris, and M. F. Kaashoek. Bandwidth-efficient management of DHT routing tables. In *Proceedings of the 2nd USENIX Symposium on Networked Systems Design and Implementation (NSDI'05)*, Boston, MA, USA, May 2005. USENIX.

[27] D. Liben-Nowell, H. Balakrishnan, and D. R. Karger. Observations on the Dynamic Evolution of Peer-to-Peer Networks. In *Proceedings of the First International Workshop on Peer-to-Peer Systems (IPTPS'02)*, volume 2429 of *Lecture Notes in Computer Science (LNCS)*. Springer-Verlag, 2002.

[28] D. Malkhi, M. Naor, and D. Ratajczak. Viceroy: A scalable and dynamic emulation of the butterfly. In *Proceedings of the 21st Annual ACM Symposium on Principles of Distributed Computing (PODC'02)*, New York, NY, USA, 2002. ACM Press.

[29] G. S. Manku, M. Bawa, and P. Raghavan. Symphony: Distributed Hashing in a Small World. In *Proceedings of the 4th USENIX Symposium on Internet Technologies and Systems (USITS'03)*, Seattle, WA, USA, March 2003. USENIX.

[30] A. Montresor, M. Jelasity, and Ö. Babaoglu. Chord on Demand. In *Proceedings of the 5th International Conference on Peer-To-Peer Computing (P2P'05)*. IEEE Computer Society, August 2005.

[31] P2PSIP. http://www.p2psip.org, 2006.

[32] Host Identity Payload. http://www.ietf.org/html.charters/hip-charter.html, 2006.

[33] J.A. Pouwelse, P. Garbacki, J. Wangand A. Bakker, J. Yang, A. Iosup, D. Epema, M.Reinders, M. van Steen, and H. J. Sips. Tribler: A social-based based peer to peer system. In *Proceedings of the 5th International Workshop on Peer-to-Peer Systems (IPTPS'06)*, February 2006.

[34] S. Ratnasamy. *A Scalable Content-Addressable Network*. PhD thesis, University of California at Berkeley, Berkeley, CA, USA, 2002.

[35] S. Rhea, D. Geels, T. Roscoe, and J. Kubiatowicz. Handling Churn in a DHT. In *Proceedings of the 2004 USENIX Annual Technical Conference (USENIX'04)*, Boston, MA, USA, June 2004. USENIX.

[36] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *Proceedings of the 2nd ACM/IFIP International Conference on Middleware (MIDDLEWARE'01)*, volume 2218 of *Lecture Notes in Computer Science (LNCS)*, pages 329–350, Heidelberg, Germany, November 2001. Springer-Verlag.

[37] T. M. Shafaat, A. Ghodsi, and S. Haridi. Handling Network Partitions and Mergers in Structured Overlay Networks. In *Proceedings of the 7th International Conference on Peer-To-Peer Computing (P2P'07)*, September 2007.

[38] I. Stoica, D. Adkins, S. Ratnasamy, S. Shenker, S. Surana, and S. Zhuang. Internet Indirection Infrastructure. In *Proceedings of the First Interational Workshop on Peer-to-Peer Systems (IPTPS'02)*, Lecture Notes in Computer Science (LNCS), pages 191–202, London, UK, 2002. Springer-Verlag.

[39] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan. Chord: a scalable peer-to-peer lookup protocol for internet applications. *IEEE/ACM Transactions on Networking (TON)*, 11(1):17–32, 2003.

[40] R. van Renesse, Y. Minsky, and M. Hayden. A Gossip-Style Failure Detection Service. In *Proceedings of the IFIP/ACM International Conference on MIDDLEWARE (MIDDLEWARE'98)*, Lecture Notes in Computer Science (LNCS). Springer-Verlag, 1998.

[41] S. Voulgaris, D. Gavidia, and M. van Steen. Cyclon: Inexpensive membership management for unstructured p2p overlays. *Journal of Network and Systems Management*, 13(2), 2005.

[42] S. Voulgaris and M. van Steen. Epidemic-Style Management of Semantic Overlays for Content-Based Searching. In *Proceedings of the 11th European Conference on Parallel Computing (EUROPAR'05)*. Springer-Verlag, 2005.