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 Jalasity 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. 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. 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, uniform 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 RSVP [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.
This page of the document contains text that discusses the use of gossip-based membership protocols in network systems, specifically in the context of self-organizing networks (SONs). The text covers the following aspects:

1. **Healing From Merged Partitions and Loopy Networks**
   - Gossip-based protocols have been successfully used to handle network partitions. The reasons for this are explained, focusing on the efficiency of these protocols in dealing with sudden state changes and maintaining efficient topology recovery.
   - There are several gossip-based algorithms that can be used to tackle both mentioned problems in SONs, such as T-chord and other protocols.
   - Membership protocols have extensively been studied in the context of gossip-based algorithms, particularly in relation to SONs. These systems exhibit desirable properties such as robustness, low clustering coefficient, and highly symmetric and similar in and out-degrees.

2. **Membership for SONs**
   - The membership requirement in SONs is strict, and the design allows for little flexibility in where nodes can be placed. The Pastry system and CAN have strict requirements for nodes and pointers, which makes it difficult to use gossip-based algorithms for membership management.
   - Researchers have suggested running gossip-based algorithms on top of SONs, but this requires addressing the strict membership requirements that SONs have.

3. **Gossiping on top of a SON**
   - Researchers have proposed using SONs as a base for gossip-based membership protocols. They note that the flexibility of such SONs makes them ideal for gossip-based membership protocols.
   - The text concludes by mentioning that while some protocols can achieve this, since gossip-based membership already has been used to construct many different types of graphs, for example, T-man can be used with a ranking function that ranks nodes according to the probability distribution described by the small-world SON, which is an ideal scenario for gossip-based membership protocols.
In contrast, Jelasity is, however, no analysis of the accuracy of such an estimate. 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, $\delta$, in the identifier space. The network size is then the ratio between the identifier space size and $\delta$. 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 ensures that some other nodes $j$ sets $v_j = v_i + v_j$. 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


