Random-walk gossip-based manycast with partition detection

Mikael Asplund, Simin Nadjm-Tehrani
Department of Computer and Information Science, Linköping University
SE-581 83 Linköping, Sweden
{mikas,simin}@ida.liu.se

Abstract

Communication is vital for successful cooperation in the event of a disaster. Such a situation calls for dissemination methods that provide partition-tolerant, energy-efficient, timely, and reliable delivery. We present a protocol that meets these requirements by combining a novel random walk gossip with a partition detection mechanism. The protocol is able to guarantee that at least M nodes will be reached by the message without having to know which nodes are in the system.

1 Introduction

When the communication infrastructure is needed the most - in the event of a disaster - it is the most likely that it is not available. We consider how rescue personnel working in such conditions can be supported by networking protocols which are tolerant to disconnectivity and unstructured topologies. Moreover, since communication devices will probably be battery driven and power is not easily available, protocols need to very restrictive in communication to save power. In particular, we are interested in reliable multicast operations in which a sender wants to send a message that can be relied upon to reach at least a portion of a certain group of receivers (i.e., manycast). Epidemic algorithms in mobile networks can be broadly categorised as using localised gossiping [2] or anti-entropy [5]. Both mechanisms have drawbacks; while the former approach suffers from a complicated balance between wasting resources and the risk of messages not being propagated, the latter provides full coverage but generally results in slow propagation as well as a high bandwidth usage. Recently, Khelil et al. [3] proposed to use a combination of these approaches called hyper-gossiping to achieve best-effort broadcasts in partitioned networks.

We believe that cooperation in post-disaster areas requires a new kind of protocol which is efficient (short delay, low bandwidth), capable of dealing with disrupted communication, reliable, and which does not require knowledge of which nodes that are in the system.

We present the basics of a protocol called RWG which is an acronym for Random Walk Gossip. We believe that this protocol meets the need for energy-efficient reliable communication in an intermittently connected environment. The protocol relies heavily on the idea to use hashing of node addresses instead of keeping track of all the nodes in the system. This way we take the middle way between best-effort algorithms requiring no knowledge at all and fully reliable protocols requiring full knowledge. The guarantee that we can provide using RWG is that at least M nodes will be reached by the message. Moreover, we expect that in a benign network, it will reach all nodes with high probability and with low latency.

2 Random walk gossip algorithm

The protocol has two modes: gossiping and waiting. During the gossiping phase, the message spreads out in the network while trying to asses whether the system is partitioned or not. If a holder of a message (custodian) decides that the network is partitioned, it puts the message on hold. This will cause nodes to be silent when no new nodes can be reached, and thus reducing energy-consumption. Eventually, the node will discover that uninfected nodes are nearby and resume propagation of the message.

As opposed to many other multicast protocols, we assume very little regarding network connectivity. The only requirement is a form of network liveness which dictates that for any two nodes n_i and n_j and time t there are nodes $n_1, \dots n_k$ and contacts $(n_i, n_1, t_1), (n_1, n_2, t_2), \dots, (n_k, n_j, t_{k+1})$ where $t < t_1 < \dots < t_{k+1}$ and a contact is an opportunity to send a message between two nodes at a given time.

Moreover, we assume that the approximate number of nodes in the system is known. *No* other knowledge is assumed such as node addresses, when contacts will occur, or geographic information. Finally, the current version of the protocol requires infinite buffers for keeping track of deleted messages. This can be traded for less strong delivery guarantees or by accepting at least once delivery seman-

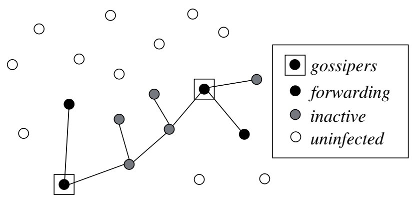

Figure 1. Gossiping phase

tics as opposed to exactly once semantics.

Each message is composed of the following fields: message id (mid, hop-count, visited nodes ($visited$), recently visited nodes ($recently$), control information, and data. The visited node field is a vector of a length which is at least M, where M is the number of nodes that must be delivered to. The semantics is the following: if $hash(mid, i) = j$ where i is a visited node, then $visited[j] = 1$, where hash() is a standard hash-function (e.g. modulo division).

The gossiping phase simulates the ideal gossip as proposed by Demers et al. [1]. This requires that a random node is selected to which to send the message (assuming push-semantics). However, since we do not know which nodes are in the system, this selection process is done by letting the message perform a random walk in the network for a number of hops. This random walk acts as a random selection mechanism.

Figure 1 shows the basic idea. The nodes marked with a square are the ones corresponding to the gossiping nodes in a normal networks. The other marked nodes will also hear the message but as a side-effect.

Algorithm 1 describes the basic behaviour during the gossipping phase of the algorithm. The mechanism to choose a random neighbour does not require neighbourhood knowledge. Instead the message is broadcasted to all 1-hop reachable nodes. Each receiver sets a timer and when the timer expires a request-to-forward is sent. The node who sent the message will only give permission to one such forward request.

Algorithm 1 Random walk gossip

```
Copy all 1s from $recently$ to $visited$ and set $recently$ to empty
Send message to a random neighbour with 0 in $recently$
When a message $m$ is heard:
  update $visited$
  decide if forwarder
After $log(n)$ hops, duplicate message and repeat
When $M$ nodes have been reached (easily seen in $visited$):
  stop forwarding and propagate $delete(m)$
```

The partition detection mechanism is invoked every time a message is received by a node. The decision to put a message in waiting is specific to that message alone, since other messages might already have seen the parts of the network which are deemed to be currently unreachable. The mechanism is described in Algorithm 2. The core idea is to compare the lists $visisted$ and $recently$, and decide the probability that the network is partitioned.

Algorithm 2 Partition detection

```
Compare the lists $visited$ and $recently$
Assume that visitations are independent
Calculate the probability of overlap
If probability less than $P_{min}$:
  put $m$ on hold
```

As an example, consider a node with the following visited vector $[0, 1, 0, 0, 1, 0, 1, 0, 1]$ and $recently = [0, 1, 0, 0, 0, 0, 1, 0, 1]$. All recently visited nodes had been visited previously. This indicates that the message is confronted with a small partition of 4-5 nodes. If, on the other hand, the recently vector would look like $[0, 1, 1, 0, 0, 0, 0, 0, 1]$ with only 2 out of three previously visited, it is too early to tell.

3 Conclusions and future work

We have sketched the basics of a protocol for disseminating data in a partitionable network where energy is scarce and transmissions should be kept to a minimum. By using hashes of addresses the protocol is able to provide absolute delivery guarantees without requiring expensive membership services. The random walk gossiping avoids the problems of flooding-type algorithms where a lot of redundant transmissions are made. We plan to continue this work by performing simulations and comparing with similar approaches such as hypergossiping [3] and route driven gossip [4], and considering more ways to optimise energy-consumption.

References

[1] 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 sixth annual ACM Symposium on Principles of distributed computing (PODC'87)*, pages 1–12, 1987. ACM Press.

[2] Z. Haas, J. Halpern, and L. Li. Gossip-based ad hoc routing. *IEEE/ACM Trans. Networking*, 14(3):479–491, 2006.

[3] A. Khelil, P. J. Marrón, C. Becker, and K. Rothermel. Hypergossiping: A generalized broadcast strategy for mobile ad hoc networks. *Ad Hoc Netw.*, 5(5):531–546, 2007.

[4] J. Luo, P. Eugster, and J.-P. Hubaux. Route driven gossip: probabilistic reliable multicast in ad hoc networks. In *Proceedings of the Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'03)*, volume 3, pages 2229–2239, 2003. IEEE.

[5] A. Vahdat and D. Becker. Epidemic routing for partially connected ad hoc networks. Technical report, Duke University, 2000.