Engineering Emergence through Gossip

Márk Jelasity*
*University of Bologna
Mura Anteo Zamboni 7, 40127 Bologna, Italy
jelasity@cs.unibo.it
and RGAI, MTA-SZTE Szeged, Hungary

Abstract

Gossip is one of the most usual social activities. The result of gossip is that new and interesting information spreads over a social network not unlike diseases during an epidemic, or computer worms over the Internet. We will argue here that the core "idea" of gossip, that is, periodic information exchange among members of a group over a network that connects them, and a subsequent update of the knowledge of the group members based on the information they exchange, is a powerful abstraction that can be applied for solving a wide range of problems in distributed computing. The applications include—apart from the most natural one: information dissemination—gathering global knowledge about distributed systems and organizing the group members into several structures, such as ordering, clustering or other arbitrary topologies.

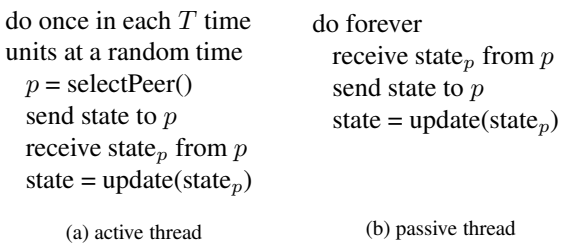
1 Introduction

Gossip is one of the most usual social activities. The result of gossip is that new and interesting information spreads over a social network very efficiently, not unlike diseases during an epidemic, or computer worms over the Internet.

The characteristics of information spreading through gossip are quite remarkable. Considering that participants only talk to their acquaintances and relatives, and they make strictly local and private decisions about what to gossip, and how to interpret the received information, it is quite impressive how efficient the process is. This fact has not been left unnoticed in the distributed algorithms community: in fact, the application of gossip to spread information over various distributed systems is commonplace, see e.g. Eugster et al. (2004).

However, the basic "protocol" underlying gossiping holds a much more general potential than merely information spreading. If we distill the basic components, we can realize that we have a complex social network that connects people and the "algorithm" which is run by all people is essentially periodic communication with some neighbors in this network. During such a communication, a person selects information to be shared with the neighbor, and receives information from the neighbor. After the reception of information, everyone updates their knowledge.

This scheme can be easily translated into the language of distributed systems, where the participants are processes or network nodes, and the social network becomes a physical or virtual (overlay) computer network. The skeleton of the gossip scheme is shown in Figure 1. Note that this scheme is rather similar to a cellular automaton, only more general in that the connection topology can be arbitrary, and it can even change over time. Furthermore, the nodes can execute rather complex algorithms to update their states. The components of the scheme are the following:

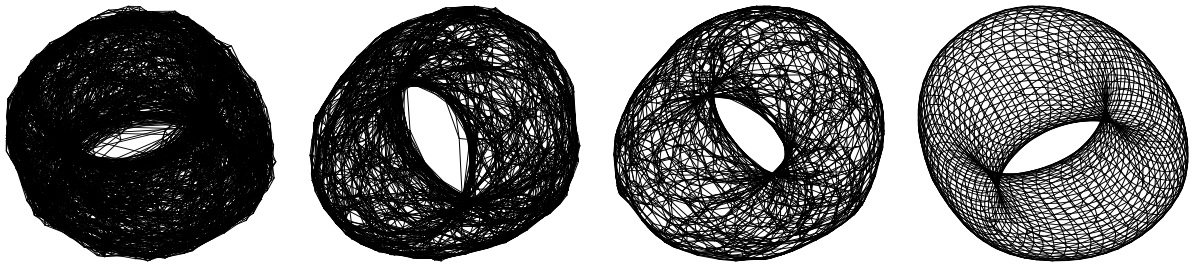


(a) active thread (b) passive thread

Figure 1: The generic protocol scheme run on each network node.

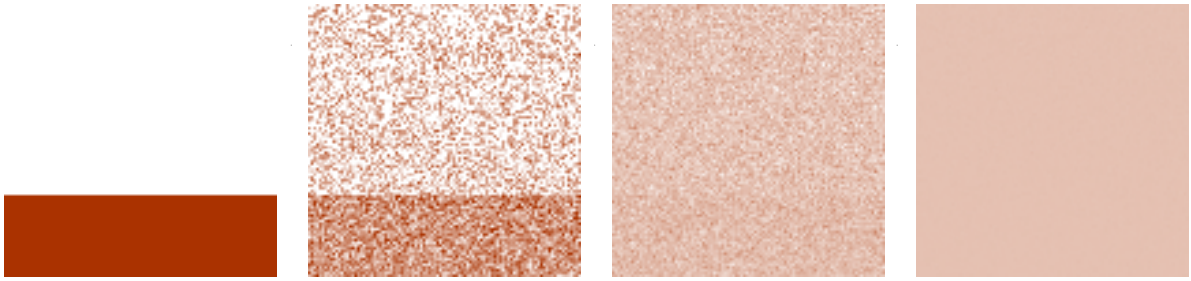
state is defined by the application domain (for example, a number, a set of documents, a set of neighbors, known information items, etc)

selectPeer() defines the way the peer is selected



after 3 cycles | after 5 cycles | after 8 cycles | after 15 cycles

Figure 2: T-Man is run starting with a random network. A torus is evolved within a few cycles. The example shown is a 1000 node graph, but experiments show that convergence time is logarithmic in network size. A cycle is T/2 time units, that is, each node communicates once on average during a cycle.



initial state | after 1 cycle | after 3 cycles | after 8 cycles

Figure 3: A network of 10 000 nodes is shown, each pixel representing a node in a 100x100 bitmap. The intensity of each pixel represents the numeric value held by a node. The underlying random overlay network is not shown. Convergence speed is similar irrespective of initial configuration and network size. A cycle is T/2 time units, that is, each node communicates once on average during a cycle.

(random, biased towards geographic proximity or high bandwidth, etc)

update() is the key function: the local rule that results in global behavior. Analogous to the update rule of cellular automata, but more general operating on arbitrary structures (states)

2 Examples

In this section we briefly outline two examples to illustrate the generality of the gossiping scheme.

2.1 Construction of Structures

Over a set of nodes connected to the Internet, one can define a so called overlay topology based on a "who-knows-whom" relation. That is, although any node can potentially communicate with any other node, to actually communicate they have to know the address of the peer node. The set of addresses known by each node define a virtual, or overlay, network.

Overlay networks have recently received increasing attention, because they are very useful in supporting distributed protocols. Applications include routing information, and clustering and sorting the nodes according to some attributes to facilitate search.

The gossip scheme is useful also to evolve such overlay topologies in a completely decentralized way, very quickly. All we need to assume is that the nodes are able to rank any set of other nodes according to preference of selecting them as neighbors. The components are implemented as follows:

state is a set of peer addresses: the partial view. The views of the nodes define the overlay topology.

selectPeer() is biased towards nodes that are "closer" according to the actual target topology to be evolved, using the preference of the nodes.

update(a,b) generates a new partial view from the two partial views a and b. It keeps those ad-

dresses from the union of a and b that are "closest" in the target topology, again, based on the preference ranking.

Figure 2 illustrates the protocol when it is used to construct a torus. The protocol has been studied by Jelasity and Babaoglu (2004), where it was shown that it is rather independent of the characteristics of the topology that we would like to generate. The cases of the ring, torus and a binary tree were shown to converge at virtually the same, logarithmic speed in the network size.

2.2 Data Aggregation

The problem of data aggregation is to provide all the nodes with global information about the distributed system in which they participate. Examples include the average or maximal value of some attribute, such as storage capacity, available bandwidth, or temperature (in sensor networks), the size of the system (number of nodes), or the variance of some attribute. Aggregation plays an important part in monitoring and control applications.

The gossip scheme offers a possibility to implement a simple but very robust and efficient averaging scheme that follows a diffusion-like dynamics. The components of the scheme have to be implemented the following way:

state is a number, representing any attribute, like temperature, free storage, available bandwidth, etc.

selectPeer() is random from the entire system, assuming an underlying random network. There are protocols that can provide this random network, such as NEWSCAST that is based on the gossiping scheme itself, see Jelasity et al. (2004).

update(a,b) defines the aggregate function to be calculated. Some examples are maximum (or minimum), where update(a, b) =max(a, b) (or min(a, b)), or any mean of the form $f^{-1}((f(x_1)+...+f(x_n))/n)$ that covers among others average ($f(x) = x$), quadratic ($f(x) = x^2$), harmonic ($f(x) = 1/x$) and geometric ($f(x) = \ln x$) means. In this case update(a,b)= $f^{-1}((f(a) + f(b))/2)$.

Figure 3 illustrates the speed at which all the nodes converge to the average value. It has been shown that the protocol is very fast and extremely robust, see Jelasity et al.; Jelasity and Montresor (2004); Montresor et al. (2004).

3 Conclusions

It has been shown that the basic scheme underlying gossiping can be efficiently used to implement very different fully distributed functions, in a controllable, robust, simple and relatively well understood way. This means that this scheme represents a way of engineering emergent properties of systems such as structure of the connectivity network, and calculation of global information.

In fact this approach can be even incorporated into a component architecture, in which a large set of services are provided by overlay networks participating in a gossip protocol, see Babaoglu et al. (2004). In this framework, overlay networks (random and structured, as shown above) are constructed and maintained, which in turn support other higher level functions such as load balancing, information dissemination, search, and aggregation.

Finally, we note that since these ideas seem to be useful and powerful in computer science engineering, the reverse question becomes also interesting: isn't it possible that real gossip also works in much richer ways than usually assumed? For example, gossip itself can change the social network it uses for spreading, and other ways of feedback are possible, like learning about certain pieces of information might change our preferences and our gossip behavior, the set of people we talk to, which in turn changes our sources of information, and so on. This dynamics might be responsible for complex emergent social phenomena.

References

Ozalp Babaoglu, Márk Jelasity, and Alberto Montresor. Grassroots approach to self-management in large-scale distributed systems. In *Pre-Proceedings of the Workshop on Unconventional Programming Paradigms (UPP'04)*, pages 165–172, Le Mont Saint-Michel, France, September 2004. invited paper, Springer LNCS volume pending.

Patrick Th. Eugster, Rachid Guerraoui, Anne-Marie Kermarrec, and Laurent Massoulié. Epidemic information dissemination in distributed systems. *IEEE Computer*, 37(5):60–67, May 2004.

Márk Jelasity and Ozalp Babaoglu. T-Man: Fast gossip-based construction of large-scale overlay topologies. Technical Report UBLCS-2004-7, University of Bologna, Department of Computer Science, Bologna, Italy, May

2004. http://www.cs.unibo.it/pub/TR/UBLCS/2004/2004-07.pdf.

Márk Jelasity, Rachid Guerraoui, Anne-Marie Kermarrec, and Maarten van Steen. The peer sampling service: Experimental evaluation of unstructured gossip-based implementations. In Hans-Arno Jacobsen, editor, *Middleware 2004*, volume 3231 of *Lecture Notes in Computer Science*, pages 79–98. Springer-Verlag, 2004. ISBN 3-540-23428-4.

Márk Jelasity and Alberto Montresor. Epidemic-style proactive aggregation in large overlay networks. In *Proceedings of The 24th International Conference on Distributed Computing Systems (ICDCS 2004)*, pages 102–109, Tokyo, Japan, 2004. IEEE Computer Society.

Márk Jelasity, Alberto Montresor, and Ozalp Babaoglu. Gossip-based aggregation in large dynamic networks. *ACM Transactions on Computer Systems*. to appear.

Alberto Montresor, Márk Jelasity, and Ozalp Babaoglu. Robust aggregation protocols for large-scale overlay networks. In *Proceedings of The 2004 International Conference on Dependable Systems and Networks (DSN)*, pages 19–28, Florence, Italy, 2004. IEEE Computer Society.