Heterogeneous Gossiping

Davide Frey, Rachid Guerraoui, Anne-Marie Kermarrec, Maxime Monod, Vivien Quema

Heterogeneous Gossiping

Davide Frey[†] Rachid Guerraoui[*] Anne-Marie Kermarrec[†]
Maxime Monod[*] Vivien Quéma[‡]

[*]EPFL, Lausanne, Switzerland
[†]INRIA Rennes-Bretagne Atlantique, France
[‡]CNRS, Grenoble, France

Gossip is effective in large-scale systems

The increasing diffusion of low-cost network connectivity has been challenging software designers to build scalable systems spanning large masses of nodes distributed over wide area networks. A number of applications, ranging from content distribution to data storage and retrieval, demand efficient techniques to disseminate and locate data in such challenging environments. Beyond the scale of these systems, developers must manage nodes with limited and heterogeneous resources that may join and leave at arbitrary times, exhibit malicious behaviors and whose characteristics often vary over time as a result of discontinuous usage patterns in the presence of multiple applications.

The need to address these challenges has motivated the work of numerous researchers and led to the development of large-scale overlay networks and data dissemination platforms able to operate in the presence of high churn. A prominent role among these systems is played by gossip-based solutions. Initially introduced as a family of algorithms for maintaining replicated database systems [9], gossip-based protocols have been applied in a wide range of settings such as data dissemination [2, 4, 11], overlay construction [6, 20, 32] and maintenance [3, 11, 22], data aggregation [17, 21, 24], failure detection [36, 37], mobile event notification [18, 29], and, more recently, also in peer-to-peer streaming applications [27, 43, 44]. In addition, specific gossip protocols were devised to support random peer selection [3, 11, 15, 22, 25, 42], or to cope with severe types of failures and attacks [7, 19, 27, 30, 31].

An appealing feature of gossip protocols is their simplicity: each node periodically exchanges messages with randomly selected nodes. No structure needs to be maintained and this makes gossip protocols very effective against failures and churn [13, 16]. In comparison, maintaining an overlay structure (e.g., multicast tree, distributed hashtable) in an environment with churn is difficult if not impossible [28]. Tree-based approaches for instance turn out to be highly vulnerable as a failed node harms its entire subtree. Multi-tree schemes have been proposed such as Splitstream [8], Chunkyspread [38] and Coolstreaming [26, 44] where reliability is enhanced by ensuring that part of the stream is disseminated through diverse paths, but this comes from free in gossip-based protocols where the neighbors of each node keep changing over time [22].

Yet, gossip is uniform

Whereas gossip protocols are effectively extremely robust, they usually do not account for heterogeneity. Nodes typically have the same number of gossip targets (fanout) and the same dissemination period. They therefore send exactly the same number of messages and are, in this sense, inherently load-balanced. However, in a large scale heterogeneous setting, this uniform distribution of load is not always appropriate. In practice, nodes frequently exhibit significant

differences in their capabilities. These may result from differences in computational power as the same application may involve small devices together with powerful machines. They may also result from differences in network connections, parts of the network may be lossy and alter the communication between certain peers, or from differences in actual host applications with some nodes running network-greedy software while others having full bandwidth available. For instance, while the penetration of the Internet into homes has greatly favored the deployment of p2p collaborative systems, bandwidth capability of edge nodes connected through ADSL links remains extremely heterogeneous. This is particularly true for the upload capacity which is of utmost importance for collaborative systems. For instance, the average upload capacity of ADSL connections in Europe is 1077kbps, while it is 899kbps in North America and only 170kbps in Africa [1].

Requiring two nodes with significant discrepancies in their capabilities to perform the same amount of work might lead to an inefficient protocol that could simply turn out to be useless for applications such as video streaming that require fast dissemination. In such applications, the data is of relatively large size and must be delivered within a given time window: the end user typically wants its stream to start as soon as possible or to be synchronized with the live stream, i.e., the buffer used for synchronization should be the smallest possible, meaning that the dissemination of data should be done as fast as possible. Note that when applied to streaming applications, the gossip protocol is usually used to forward stream-packet identifiers, the content being subsequently pulled whenever necessary. Even for peer-to-peer file sharing, the end user may want its application to deliver selected files as early as possible, i.e., downloading them as fast as possible. Besides inefficiency, a uniform gossip protocol may appear unfair in a heterogeneous setting and can be frustrating even if nodes indeed have the same capabilities but some benefit significantly more than others from the dissemination. For instance, in a news-oriented application, some nodes may subscribe to fewer topics than others and thus should also receive fewer events.[1] With a uniform gossip protocol, these nodes may find it unfair to contribute to the system as much as the other nodes and may thus decide to leave the application. Again, this calls for a gossip scheme that accounts for heterogeneity. In fact, many researchers have recognized this need in some form or another [5, 10, 33, 35, 38–41].

Breaking gossip uniformity is challenging

Intuitively, a gossip protocol could account for heterogeneity by introducing a bias towards *wealthy* nodes so that they perform more work. Two approaches can be considered:

1. Increasing the fanout of wealthy nodes. This basically means that wealthy nodes will send more messages in a given gossip round.[2]

2. Increasing the probability that a wealthy node is selected in a gossip round. This basically means that wealthy nodes will contribute to more gossip rounds.

Putting these ideas to work is however challenging and a mechanism that accounts for heterogeneity should satisfy a number of non-trivial constraints.

[1] Within the sport news system of the Télévision Suisse Romande for instance, some users (nodes) subscribe to results about soccer or hockey, which indeed generate a lot of information, while others only subscribe to results about cycling, which generates less traffic, or in chess, which generates very few events in Switzerland. While appealing, a gossip-based implementation of such a system might lead to a feeling of unfairness if the dissemination of sport results is uniformly spread among all nodes.

[2] Biasing the frequency of dissemination has the same effect.

- *Adapting real contribution.* At first glance, each of the two approaches suggested above effectively ensures that wealthy nodes will contribute more to the dissemination by sending more messages. However, the number of messages might not be a good metric for the real contribution of a node. This is particularly the case in a gossip-based video-streaming application. In such a setting, two kinds of messages are typically considered. Those that simply contain stream ids (fairly cheap) and those that contain actual streams. A node first receives an id and requests the actual stream if it does not have it. Sending the actual stream clearly introduces a higher overhead than sending an id. Hence, a mechanism that accounts for heterogeneity should have wealthy nodes really perform more work and not only send more messages.

- *Preserving some randomness.* The reliability of gossip protocols stems from the randomness of the dissemination process. Any bias toward more powerful nodes might hamper this randomness with the risk of decreasing reliability. Relying on a subset of wealthy nodes to work more may, in fact, have a negative impact in case many of these nodes leave the system or, even worse, behave maliciously.

- *Local knowledge of global capability.* In a large scale setting, one cannot assume global knowledge of the capabilities of all nodes for the same reason for which membership protocols manage partial views instead of global views [11]. Global knowledge of capabilities is unattainable because of churn and dynamic capability changes as we discuss below. On the other hand, nodes cannot adapt their workloads based solely on local information about their own capabilities. For instance, assume that each node systematically decreases its fanout by a certain percentage as soon as its capability decreases by the same percentage, then a sudden decrease in the capability values of a number of nodes is likely to interrupt the dissemination process.

- *Adapting to capability changes.* Capability usages dynamically change over time as applications use bandwidth depending on user behavior (e.g., reading emails, downloading files or watching online multimedia content) and on the peer-to-peer offer (e.g., changes in the availability of files and uploaders). For example, a network-greedy application for peer-to-peer file sharing stops using download bandwidth as soon as the transfers of requested files have completed, which for many people means the application can be closed resulting in a stop of upload bandwidth usage. Hence, a mechanism that accounts for heterogeneity should itself be dynamic. Furthermore, the time required to account for the change should be smaller than the stability period of capabilities.

Gossip theory to the rescue

In the attempt to address these challenges, we have recently proposed HEAP (*HEterogeneity-Aware Gossip Protocol*) [14], a protocol for collaborative, high-bandwidth content distribution.

Two observations lie at the heart of HEAP:

1. Mathematical and empirical results [12, 34] convey the fact that reliability is ensured as long as the *average* of all fanouts is approximately $\ln(n)$ (in a system of size n). This suggests a way to tune the contributions of nodes by adapting their fanouts according to their capabilities without impacting the reliability of dissemination.

2. The utility of a message is highest in the *first* hops of dissemination. Thus the capability of a node intervening at the first gossip rounds can have a significant impact on performance [23]. This suggests ways to bias the selection of gossip targets so that the positions

of nodes in the data-forwarding process match their capabilities. For example, nodes with a very high bandwidth or that are very responsive should be involved at the earliest stages of dissemination in order to optimize the average latency experienced by data receivers. This is particularly crucial in streaming applications where significant work is performed in the early stages of the dissemination where all streams are required.

HEAP incorporates a gossip-based aggregation protocol [21, 36] through which each node may obtain a very good sample of the overall system capabilities. This allows nodes *(i)* to adapt their fanout values based on their capabilities while maintaining the average fanout constant across the whole system; and *(ii)* to bias the selection of their gossip targets so that the most capable nodes are chosen at the first stages of dissemination. This allows HEAP to optimize the distribution of nodes in the dissemination chain ultimately increasing the experienced performance.

HEAP also incorporates LIFT *(LIghtweight Freerider Tracking)*, a companion subprotocol able to track selfish nodes that declare high capability values in order to augment their perceived performance, without contributing accordingly. Although interesting in its own right, LIFT is particularly important in the context of HEAP as biasing the selection of gossip targets may render the protocol more vulnerable to the presence of nodes that do not contribute as they should.

References

[1] `http://www.speedtest.net`.

[2] *Gossip-based computer networking*, volume 41. 2007.

[3] A. Allavena, A. Demers, and J. E. Hopcroft. Correctness of a gossip based membership protocol. In *Proceedings of the 24th symposium on Principles of distributed computing (PODC)*, 2005.

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

[5] M. Bishop, S. Rao, and K. Sripanidulchai. Considering priority in overlay multicast protocols under heterogeneous environments. In *Proceedings of the 25th Conference on Computer Communications (INFOCOM)*, 2006.

[6] F. Bonnet, A.-M. Kermarrec, and M. Raynal. Small world networks: From theoretical bounds to practical systems. In *Proceedings of the 11th International conference on principles of distributed systems (OPODIS)*, 2007.

[7] E. Bortnikov, M. Gurevich, I. Keidar, G. Kliot, and A. Shraer. Brahms: Byzantine resilient random membership sampling. In *Proceedings of the 27th symposium on Principles of distributed computing (PODC)*, 2008.

[8] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh. SplitStream: high-bandwidth multicast in cooperative environments. In *Proceedings of the 19th symposium on Operating systems principles (SOSP)*, 2003.

[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 6th symposium on Principles of distributed computing (PODC)*, 1987.

[10] M. Deshpande, B. Xing, I. Lazardis, B. Hore, N. Venkatasubramanian, and S. Mehrotra. Crew: A gossip-based flash-dissemination system. In *Proceedings of the 26th International Conference on Distributed Computing Systems (ICDCS)*, 2006.

[11] P. T. Eugster, R. Guerraoui, S. B. Handurukande, A.-M. Kermarrec, and P. Kouznetsov. Lightweight probabilistic broadcast. *ACM Transactions on Computer Systems*, 21(4):341 – 374, 2003.

[12] P. T. Eugster, R. Guerraoui, A.-M. Kermarrec, and L. Massoulié. Epidemic information dissemination in distributed systems. *IEEE Computer*, 37(5):60–67, 2004.

[13] P. T. Eugster, R. Guerraoui, and P. Kouznetsov. Δ-reliable broadcast: A probabilistic measure of broadcast reliability. In *Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS)*, 2004.

[14] D. Frey, R. Guerraoui, A.-M. Kermarrec, M. Mogensen, M. Monod, and V. Quéma. Gossiping Capabilities. Technical report, 2008.

[15] A. J. Ganesh, A.-M. Kermarrec, and L. Massoulié. Peer-to-peer membership management for gossip-based protocols. *IEEE Trans. Comput.*, 52(2):139–149, 2003.

[16] B. Godfrey, S. Shenker, and I. Stoica. Minimizing churn in distributed systems. In *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM)*, 2006.

[17] I. Gupta, R. van Renesse, and K. P. Birman. Scalable fault-tolerant aggregation in large process groups. In *Proceedings of the 2nd International Conference on Dependable Systems and Networks (DSN)*, 2001.

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

[19] M. Haridasan and R. van Renesse. Defense against intrusion in a live streaming multicast system. In *Proceedings of the 6th International Conference on Peer-to-Peer Computing (P2P)*, 2006.

[20] M. Jelasity and Ö. Babaoglu. T-man: Gossip-based overlay topology management. In *Proceedings of the 3rd International Workshop on Engineering Self-Organising Systems (ESOA), Revised Selected Papers*, 2005.

[21] M. Jelasity, A. Montresor, and Ö. Babaoglu. Gossip-based aggregation in large dynamic networks. *ACM Transactions on Computer Systems*, 23(3):219–252, 2005.

[22] M. Jelasity, S. Voulgaris, R. Guerraoui, A.-M. Kermarrec, and M. van Steen. Gossip-based peer sampling. *ACM Trans. Comput. Syst.*, 25(3):8, 2007.

[23] R. Karp, C. Schindelhauer, S. Shenker, and B. Vocking. Randomized rumour spreading. In *Proceedings of the 41st Symposium on Foundations of Computer Science (FOCS)*, 2000.

[24] D. Kempe, A. Dobra, and J. Gehrke. Gossip-based computation of aggregate information. In *Proceedings of the 44th Symposium on Foundations of Computer Science (FOCS)*, 2003.

[25] J. Leitão, J. Pereira, and L. Rodrigues. Hyparview: a membership protocol for reliable gossip-based broadcast. In *Proceedings of the 37th International Conference on Dependable Systems and Networks (DSN)*, 2007.

[26] B. Li, Y. Qu, Y. Keung, S. Xie, C. Lin, J. Liu, and X. Zhang. Inside the new coolstreaming: Principles, measurements and performance implications. In *Proceedings of the 27th Conference on Computer Communications (INFOCOM)*, 2008.

[27] H. C. Li, A. Clement, E. L. Wong, J. Napper, I. Roy, L. Alvisi, and M. Dahlin. BAR gossip. In *Proceedings of the conference on Operating Systems Design and Implementation (OSDI)*, 2006.

[28] J. Liang, S. Y. Ko, I. Gupta, and K. Nahrstedt. Mon: On-demand overlays for distributed system management. In *Proceedings of the 2nd Workshop on Real, Large Distributed Systems (WORLDS)*, pages 13–18, 2005.

[29] J. Luo, P. T. Eugster, and J.-P. Hubaux. Route driven gossip: Probabilistic reliable multicast in ad hoc networks. In *Proceedings of the 22nd Conference on Computer Communications (INFOCOM)*, 2003.

[30] D. Malkhi, E. Pavlov, and Y. Sella. Optimal unconditional information diffusion. In *Proceedings of the 15th International Conference on Distributed Computing (DISC)*, 2001.

[31] Y. M. Minsky and F. B. Schneider. Tolerating malicious gossip. *Distributed Computing*, 16(1):49–68, 2003.

[32] A. Montresor, M. Jelasity, and Ö. Babaoglu. Chord on demand. In *Proceedings of the 5th International Conference on Peer-to-Peer Computing (P2P)*, 2005.

[33] J. Pereira, L. Rodrigues, A. Pinto, and R. Oliveira. Low latency probabilistic broadcast in wide area networks. In *Proceedings of the 23rd International Symposium on Reliable Distributed Systems (SRDS)*, 2004.

[34] B. Pittel. On spreading a rumor. *SIAM Journal on Applied Mathematics*, 47(1):213–223, 1987.

[35] Y.-W. Sung, M. Bishop, and S. Rao. Enabling contribution awareness in an overlay broadcasting system. In *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM)*.

[36] R. van Renesse, K. P. Birman, and W. Vogels. Astrolabe: A robust and scalable technology for distributed system monitoring, management, and data mining. *ACM Transactions on Computer Systems*, 21(2):164–206, 2003.

[37] R. van Renesse, Y. Minsky, and M. Hayden. A gossip-style failure detection service. In *Proceedings of the 1st International Conference on Distributed Systems Platform and Open Distributed Processing (Middleware)*, 1998.

[38] V. Venkataraman, K. Yoshida, and P. Francis. Chunkyspread: Heterogeneous unstructured tree-based peer to peer multicast. In *Proceedings of the 14th International Conference on Network Protocols (ICNP)*, 2006.

[39] S. Verma and W. T. Ooi. Controlling gossip protocol infection pattern using adaptive fanout. In *Proceedings of the 25th International Conference on Distributed Computing Systems (ICDCS)*, 2005.

[40] V. Vishnumurthy and P. Francis. On heterogeneous overlay construction and random node selection in unstructured p2p networks. In *Proceedings of the 25th Conference on Computer Communications (INFOCOM)*, 2006.

[41] V. Vishnumurthy and P. Francis. A comparison of structured and unstructured p2p approaches to heterogeneous random peer selection. In *Usenix Annual Technical Conference*, 2007.

[42] S. Voulgaris, D. Gavidial, and M. van Steen. Cyclon: Inexpensive membership management for unstructured p2p overlays. *Journal of Network and Systems Management*, 13(2):197–217, 2005.

[43] M. Zhang, Q. Zhang, L. Sun, and S. Yang. Understanding the power of pull-based streaming protocol: Can we do better? *IEEE Journal on Selected Areas in Communications*, 25(9):1678–1694, 2007.

[44] X. Zhang, J. Liu, B. Li, and T.-S. P. Yum. Coolstreaming/DONet: A data-driven overlay network for efficient live media streaming. In *Proceedings of the 24th Conference on Computer Communications (INFOCOM)*, 2005.