

Gossip Based Streaming

Xinyan Zhang

Department of Information Engineering
The Chinese University of Hong Kong
Shatin, N.T., Hong Kong
xyzhang2@ie.cuhk.edu.hk

Jiangchuan Liu

Department of Computer Science and Engineering
The Chinese University of Hong Kong
Shatin, N.T., Hong Kong
ljc@cse.cuhk.edu.hk

ABSTRACT

In this paper, we propose a novel multicast streaming protocol for overlay networks, called Gossip Based Streaming (GBS). In GBS, streaming contents are not come from a single upstream source, but delivered from several sources to a client. Though being similar to existing gossip protocols, the unique requirements for streaming, such as continuous playback, are addressed in our design. Preliminary results show that GBS performs much better in dynamic user environments.

Categories and Subject Descriptors

C.2.0 [Computer Systems Organization]: Computer-Communication Networks—general

General Terms

Algorithms, Measurement, Performance, Design

Keywords

Streaming, overlay networks, multicast

1. INTRODUCTION

Streaming media through overlay networks has received much attention recently [1]-[5]. In this paper, we present a novel multicast streaming framework, *Gossip Based Streaming* (GBS), in which an overlay is construct based on a random graph topology, and streaming contents are not come from a single upstream source, but delivered from several sources to a client, like in gossiping protocols [6,7]. This is quite different from many existing tree-based overlays, as shown in Figure 1.

The principle of distributed gossip communication is simple: in each time slot, a node in the network (randomly) selects another node as a communication partner, and exchanges information with each other. Thus, over the time, contents will spread throughout the network in an epidemic fashion. A key challenge for gossiping based streaming, however, is to minimize delay and data outage, as continuity is important in a stream playback.

2. GBS OPERATIONS

GBS employs an unstructured overlay (random graph) with a bootstrapping node. A system diagram of each GBS node is shown in Fig. 2. When a new node joins the system, it first contacts the bootstrap node to obtain a candidate list of neighboring nodes. The new node then contacts these candidates to explore other nodes, finally to set its *neighbors* in the overlay. After join-

ing the overlay, a node will continuously probe other nodes to maintain a set of best neighbors, in general, of size 3 to 4.

The procedure of streaming for this node begins from establishing connections to the neighbors. We assume that a video content is divided into blocks of a fixed-size. Once connected, a node will fetch a buffer map from each of its neighbors, which indicates the availability of each block in the neighbor.

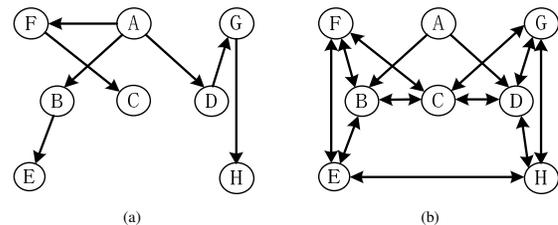


Fig. 1. Overlays structures.
(a) Application-layer multi-cast; (b) GBS.

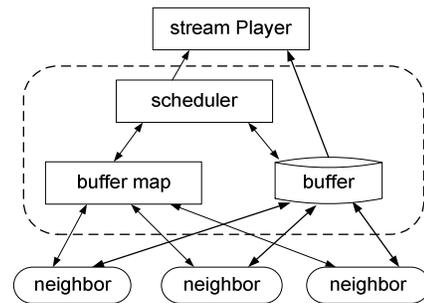


Fig. 2. System diagram for GBS.

After that, the node's scheduler continuously makes decisions on which block is to be fetched from which neighbor. A simple scheduling algorithm is given as below, where the demanded blocks is transferred from the neighbors in a round-robin fashion. This enables balanced data delivery among the neighbors. Moreover, during streaming, a node should also notify all its neighbors about its buffer status changes, which ensures that a node is aware of the up-to-date buffer status of its neighbors. Such updates can be piggybacked in data blocks to reduce network traffic.

Scheduling algorithm at a node

If local buffer is not full:

1. Find neighbor *A* with the longest available time (no data fetched from it);
 2. Find block *s* with lowest sequence number, which is available at *A* but not in the local buffer;
 3. Fetch block *s* from neighbor *A*;
-

3. ANALYTICAL RESULTS

In a dynamic environment, each node could fail or leave at will. The failure (or leave) may cause data outage in some related nodes before new delivery paths are formed. For streaming applications, the outage is highly undesired because it generally leads to discontinuity in playback. In this section, we sketch our analytical results of the outage rate for overlay networks.

We first consider a tree-based application-layer multicast, which serves as a baseline in our performance evaluation. In such an overlay, each node has up to m children, but has only one parent from which it can receive streaming contents. Assume each direct downstream node of the failure node has probability P_o of not finding a new upstream node within Δt time, it can be calculated that the average outage, *i.e.*, the number of nodes suffer from data outage in Δt time, is (see [6] for details):

$$E(D_t) = \frac{m^h - m^2}{m - 1} + (n - \frac{m^h - 1}{m - 1})(1 - (1 - P_o R \Delta t)^{h-1}) - \frac{m^2(1 - P_o R \Delta t)(m(1 - P_o R \Delta t))^{h-2} - 1}{m(1 - P_o R \Delta t) - 1}$$

where h is the height of the multicast tree, R is the average failure rate of node.

In GBS, however, each node has m neighbors and all of them can provide streaming contents to the node, which potentially lead to much better robustness. Based on a random graph model, we can calculate the data outage rate $E(D_g)$ for GBS as

$$E(D_g) = nP_o((1 - P_s + P_s R \Delta t)^m - ((1 - P_s)(1 - R \Delta t))^m)$$

where P_s is the probability that a node itself can support streaming to all its neighbors, *i.e.*, serving as a parent node like in the tree case. Both P_o and P_s depend on network delay and bandwidth, and can be estimated given any specific configurations.

4. SIMULATION RESULTS

We have conducted several preliminary experiments to investigate the performance of GBS. Fig. 3 shows the node data outage as a function of the overlay size. For comparison, we also show the analytical results as well as the results for tree-based application-layer multicast, which operates as follows: when a new node joins the overlay, it first finds the oldest node that can accept a new child (less than m children) and then retrieves the streaming content from the node.

It can be seen that the average node data outage of GBS is much lower than that of the tree-based application-layer multicast. The difference is particularly remarkable for large overlays. As an example, for an overlay of 5000 nodes, the data outage of GBS (2.5%) is nearly an order of magnitude lower than that of tree (20%). This implies that GBS is more scalable in dynamic environment than a tree based structure.

5. SUMMARY AND FUTURE WORK

This paper has presented the basic design of Gossip Based Streaming (GBS). We have sketched some analytical results, particularly for the data outage rate, which matches well with the simulation results. More importantly, both simulation and analytical results reveal that GBS performs much better than tree-based overlays in dynamic environments, where nodes could fail or leave at will. We are currently improving the GBS framework along three directions:

1. Enhancing the operations in GBS; as an example, we are developing better scheduling algorithms for retrieving blocks from neighbors;
2. Analyzing the transmission delay in GBS, as streaming applications are generally delay-sensitive; some preliminary results can be found in [9]
3. Performing simulation under different network configurations, and building a prototype based on the PlanetLab platform, in which we are currently participating.

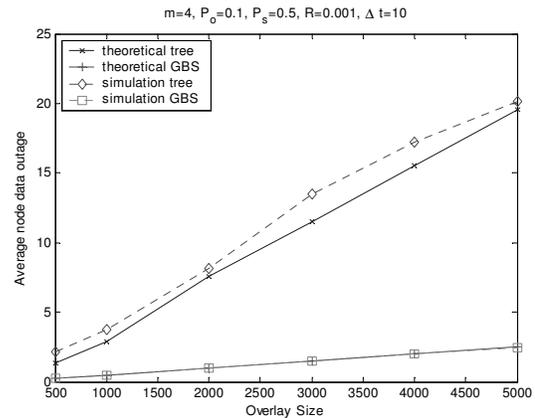


Fig. 3. Data outage as a function of overlay size. The Y-axis is the average percentage of nodes that suffer from data outage in a period of Δt seconds.

6. ACKNOWLEDGMENT

This work was supported by grants from the Research Grants Council of the Hong Kong Special Administrative Region, China, under Grant CUHK C001/2050312.

7. REFERENCES

- [1] X. Zhang, Q. Zhang, Z. Zhang, G. Song, W. Zhu, "A Construction of Locality-Aware Overlay Network: mOverlay and its performance," *IEEE JSAC Special Issue on Recent Advances on Service Overlay Networks*, Jan. 2004.
- [2] X. Zhang, G. Song, Q. Zhang, and W. Zhu, "Performance Analysis in Unstructured Overlays", in *Proc. IEEE ICC*, May 2003.
- [3] M. Castro *et al.*, "SplitStream: High-bandwidth multicast in cooperative environments", in *Proc. ACM SOSP*, Oct. 2003.
- [4] V. N. Padmanabhan, H. J. Wang, P. A. Chou, and K. Sripanidkulchai, "Distributing streaming media content using cooperative networking," in *Proc. NOSSDAV*, May 2002.
- [5] S. Banerjee, S. Lee, B. Bhattacharjee, and A. Srinivasan, "Resilient multicast using overlays," in *Proc. ACM SIGMETRICS*, June 2003.
- [6] D. Kempe, J. Kleinberg, A. Demers. "Spatial gossip and resource location protocols." in *Proc. 33rd ACM Symposium on Theory of Computing*, 2001.
- [7] A.J. Ganesh, A.-M. Kermarrec, and L. Massoulié, "Peer-to-peer membership management for gossip-based protocols," to appear in *IEEE Trans. Computers*, Feb. 2003.
- [8] X. Zhang and J. Liu, "Gossip based streaming," *Technical Report (draft)*, Feb. 2004.