Interactive Scalable Crowdcasting

Shun-Yun Hu
Institute of Information Science
Academia Sinica, Taiwan, R.O.C.
syhu@iis.sinica.edu.tw

***Abstract*—Supporting real-world activities online has been one of the main goals in applied computer science. However, the gathering of a large crowd for a specific purpose (e.g., lecture, concert, rally), has yet been able to move online. The recent concept of crowdcasting (i.e., broadcasting to an audience while engaging them with real-time feedback) may pave the way towards it. However, the audience scale and feedback fidelity are still limited by bandwidth and latency constrains. Increasing the availability of such resources thus can be important to support new forms of online activities.**

In this concept paper, we propose IMON, a scalable approach to support crowdcasting where the number of participants can be potentially large, while the information content delivered can be rich. IMON is a peer-to-peer (P2P) overlay multicast that relies on idle peer bandwidth, and self-organizing structures to support scalable crowdcasting. The main design and challenges of IMON are described for future evaluations.

I. Introduction

One important human activity is the gathering for a particular purpose. From family to concerts or festivals, gathering provides important social bonding that helps to shape identities and cultures. Large-scale gatherings (e.g., lectures, concerts, fairs, parades, or New Year Eve's countdown) are particularly significant as many can be involved in a rich experience, such that both a single message (by a speaker) and local interactions (with nearby friends) are available.

However, gathering of a big crowd has been difficult to replicate online, though continuous progress has been made towards this direction, such as the recent concept of *crowdcasting*[1]. The basic idea is to combine both *push* (i.e., broadcasting) and *pull* (i.e., crowdsourcing) to create a more engaging user experience. In other words, broadcasting certain messages to an audience, while getting feedback from them in some way. For example, online radio stations can elicit real-time audience votes about what songs to play next. However, as the technique is still at its infancy, audience feedback is little more than small text messages, while supportable audience is also be capped by the bandwidth of the broadcaster.

We are interested to see whether crowdcasting can be extended to a specific form of large-scale gathering, namely, big lectures and radio shows, where a single speaker (or at most a few invited panelists) could speak to a large audience during a *talk stage*, while anyone from the audience can ask questions or signal intentions during an *interactive stage*. To mimic real-world events, it is also desirable if: 1) the fidelity of the broadcast can be high (e.g., video or even gesture streams can be supported); 2) there is no limit to the live participant size; 3) audience can participate with high fidelity feedback (e.g., voice or video stream, not just text).

We tackle this problem by starting with a few observations: 1) fidelity and scalability are essentially the same problem of lacking enough sender bandwidth; 2) during the talk stage, latency only matters for those who are interacting (e.g., among the panelists), but some delays may be acceptable for the rest, as long as the streaming is continuous; 3) latencies can be seen also *as a bandwidth problem*, as if the bandwidth is sufficient, only end-to-end latency exists (and no queuing delays).

Our approach involves two main components: the first is to find an effective method for *bandwidth amplification*, that is, increasing the overall bandwidth capacities at the sender's side. The second is to connect the audience into an *latency-aware multicast tree*, so audience may join scalably, while keeping latencies within reasonable limits. The central assumption behind is that Internet-connected machines are often idle, with excess CPU and bandwidth. If utilized effectively, bandwidth amplification and latency-aware multicasting, may become possible (e.g., as shown by SETI@Home and Skype [1])

II. Design of IMON

The basic idea behind IMON (Interactive Multicast Overlay Network) is that, by assembling potential audience machines (called *peers* or *nodes*) into a peer-to-peer (P2P) network and utilizing idle bandwidth, it is possible to increase per-peer download bandwidth to support high fidelity multicasting. Scalability is achieved by chaining the peers into a self-organizing multicast tree resilient to host join or leave (i.e., churning). Finally, interactivity is supported by having direct communications with the multicast source node, but only a limited number of interacting users are allowed at any time, to avoid overloading the source or the audience.

We assume that there are many idle peers such that their resources can be utilized. If each peer can be categorized as either *active* or *idle*, we assume that the number of idle peers is a multiple (at least 2) of the active peers in any given crowdcast. We describe IMON's two main aspects below:

Bandwidth Amplification For transmissions on the Internet, there is often a *critical path* from the source to destination that limits the bandwidth available (such can occur either because of bandwidth capping or throttling at the last-mile ISPs, or due to congestions on the transmission path[2]). However,

[1]http://en.wikipedia.org/wiki/Crowdcasting

[2]http://en.wikipedia.org/wiki/Bandwidth_throttling

closer to the sender or receivers, there may exist hosts that could have higher throughput with either the sender or receiver. The idea behind bandwidth amplification thus is simple: send *fragments* of the intended transmission first to some *proximity relays* (i.e., relay nodes that are closer to the source or destination latency-wise) who have enough bandwidth, then allow the proximity relays to deliver the packets to the destination in higher throughput. Some initial delays will incur due to the initial separation of the transmission, however, as long as the throughput between the relays and the transmitting nodes can be larger than the original direct path, then bandwidth amplification is possible. For example, in Fig. 1, assuming that source node A and destination node B has a transmission of only 5 bytes / second, yet between A and each of the relays, a 5 bytes/second bandwidth exist (so is between the relays and node B), then node A can first transmit packet fragments to the relays first, in a pipelining fashion, before the relays send higher-bandwidth transmission to node B. Note that we assume that an overall higher-bandwidth connections can be established between node A or B and the relays.

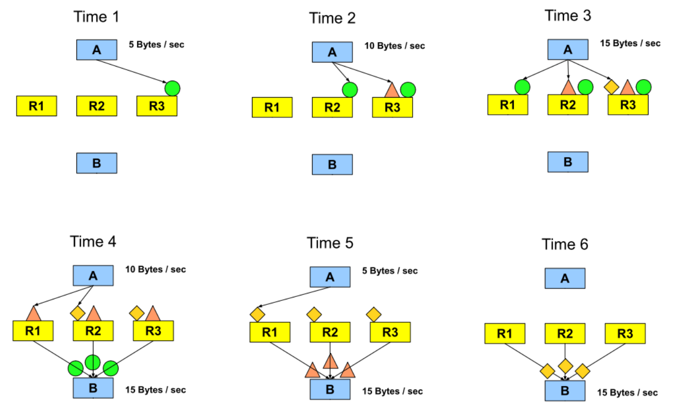

Fig. 1. Sequence of how node A's transmission to node B can be amplified via three relays R. The squares, triangles and circles represent packet fragments of 5 bytes each. Different shapes represent different packets.

Latency-aware Multicast Tree In IMON, *all* nodes of the system on the network would join the same overlay, regardless of their particular interests, to allow the sharing of idle resources. Considering that nodes often have different capacities in CPU or bandwidth, the backbone of the overlay is formed by only *supernodes* who are capable (i.e., have enough resources) and publicly reachable (i.e., not behind a NAT). Other regular nodes would find the the closest available supernode to connect in forming the network. All joining nodes first identify themselves with a *physical coordinate* (e.g., network coordinates such as Vivaldi [2]), such that the distance between two coordinates reflects the latency between the nodes. All nodes thus join the network with a coordinate point, with the supernodes in particular forming a Voronoi overlay [3]. The reason for a Voronoi overlay is that messages can be forwarded from any given source node to all other nodes within a specified radius, by constructing a unique spanning-tree covering all nodes within the radius [3]. This tree can be constructed on-the-fly without centralized control. As such, the failure of any given node on the path, can also be quickly replaced by alternative paths via other nodes.

There are two remaining issues: 1) how to ensure that only users relevant to a particular broadcast would receive the relevant messages? and 2) how can the latency be controlled? For the first issue, a basic solution is for each node to join two overlays: the *physical overlay* and the *channel overlay*. The coordinates of a node in both overlays are the same, but the membership in each overlay is different. The physical overlay consists of all nodes in the system, so that the overlay neighbors are the physically closest (i.e., with the least latency). Channel overlay on the other hand, consists of only nodes interested in a given multicast channel. To join either overlay, a joining node simply contacts any existing node on the respective overlay, and performs greedy forwarding to its closest neighbors [3]. However, for the channel overlay, each joining node will have to first contact the channel-starting *source node*, which can be queried from either a central server (for faster query), or a Distributed Hash Table (DHT) (for better fault tolerance). Before joining the channel overlay, each node needs to first join the physical overlay, and query its neighbors on the physical overlays to find a set of *proximity relays*. The proximity relays will then join a particular channel overlay on behalf of the subscribing node.

Once a channel overlay is roughly constructed, a spanning tree rooted at the channel source can be built on top of the overlay, with the desirable property that the tree-depth is minimized. To ensure that latency is bounded, one simple heuristics is to start with a spanning tree that covers all the proximity relays by using techniques such as VoroCast [3]. Then, a pruning process begins by joining certain branches of the tree to an *accommodating node* closer to the source node, provided that the accommodating node has enough upload bandwidth. This pruning process can be iteratively run from the bottom of the multicast tree to the source, with the goal to minimize tree depth. Note that in this design, nodes involved in a channel overlay are not necessarily interested in the channel's content, as they may simply serve as the proximity relays.

To allow participation during the *interactive stage*, a few nodes selected will connect and send their packets directly to the channel source. After mixed with other participants' packets, the mixed packets can then be delivered over the channel overlay similar to other multicast packets.

We have described the design of a scalable crowdcast system. By using idle bandwidth beyond channel participants, a spanning tree with minimal depth can be built to deliver higher bandwidth streaming. We will evaluate IMON with a focus on tree pruning methods and bandwidth amplification.

References

[1] S. Guha, N. Daswani, and R. Jain, "An experimental study of the skype peer-to-peer voip system," in *Proc. IPTPS*, 2006.
[2] F. Dabek, R. Cox, F. Kaashoek, and R. Morris, "Vivaldi: a decentralized network coordinate system," in *Proc. SIGCOMM*, 2004.
[3] J.-R. Jiang, Y.-L. Huang, and S.-Y. Hu, "Scalable aoi-cast for peer-to-peer networked virtual environments," in *ICDCS Workshops*, 2008.