FLaCoSt: A Novel Peer to Peer Architecture for Video Streaming in a Next Generation Network

Jaime Garcia-Reinoso*, Alex Bikfalvi†, Ivan Vidal* and Francisco Valera*
*Universidad Carlos III de Madrid
Avda. de la Universidad 30, 28911, Leganes (Madrid), Spain
Email: {jgr, ividal, fvalera}@it.uc3m.es
†IMDEA Networks
Avda. Mediterraneo 22, 28918, Leganes (Madrid), Spain
Email: alex.bikfalvi@imdea.org

***Abstract*—This paper proposes a peer-to-peer video streaming delivery system within the framework of Next Generation Networks using Application Level Multicast. Increased efficiency and reliability is achieved in two ways. First, the system uses path diversity by splitting the video traffic in several stripes and distributing each stripe via different multicast trees. Second, the proposed architecture lies within a Next Generation Network framework that enables network resource reservations with different classes of service. Therefore, the video stripes can receive different priorities making possible the usage of Layered Coding techniques such as Multi-Description Coding and Scalable Video Coding. The peer-to-peer protocol takes advantage of the multipath delivery and resources reservation by optimising the searches for a parent in the multicast tree, thus minimising the service access time and service interruptions. These ideas can be improved using always-on central devices like Residential Gateways as peer nodes.**

***Keywords*-Peer-to-peer; Residential Gateway; Next Generation Network; Community Network; Application Level Multicast**

I. Introduction

This article proposes a peer-to-peer (P2P) protocol for video streaming delivery within the framework of Next Generation Networks. Several P2P video applications are based on algorithms capable of defining a multicast tree that is used to distribute video between the different participants (see [1], [2]). Since P2P networks are prone to service interruptions, there are two methods that can be proposed in order to increase their reliability: *QoS in a multicast tree* and *multiple multicast trees*.

The first method uses service quality indicators (such as available bandwidth), to build an optimal multicast tree. Thus, the network guarantees the quality of service for the video path. With the second method the video stream is divided into several *stripes*, which are then forwarded using different multicast paths. This method can be used with multi-description coding algorithms making a single stripe enough to watch the video and with an increasing quality as more stripes are received ([3], [4]).

FLaCoSt (Fair Layered Coding Streaming) merges both alternatives into a single solution. The end-to-end quality of service is guaranteed by a Next Generation Network infrastructure like the one proposed in [5], where link capacity and forwarding resources can be reserved for a particular flow using SIP (many applications only provide optimisations in terms of quality without any guarantees). Although many implementation solutions are possible, this paper focuses on an environment where the entire functionality is enabled in the users' residential gateway (RGW), which become the nodes in the multicast trees. This approach not only makes the process transparent to the user, but it can also enable operators to reserve uplink capacity for deliver the video content to other users. Because the residential network resources are taken into account when the reservation is made for a particular class of service, the quality is consequently improved [6].

The rest of the article is organised as follows: Section II gathers the main contributions of this paper: the peer-to-peer architecture, protocol specifications, multicast operations and the resource reservation function. Section III evaluates with both simulations and a real implementation some performance aspects of the proposal and Sec. IV concludes this paper.

II. FLaCoSt Proposal

A. Overview

FLaCoSt intends to support multi-path video streaming scenarios in a QoS enabled environment such as NGN. The video stream is divided into different stripes (using some layered coding technique), each containing a part of the full quality. Because some stripes carry more important information than others, a set of priorities is established guaranteeing that the delay and the jitter remains under some specified limits: a high priority (HP), a medium priority (MP) and a low priority (LP). More stripes/priorities are possible, but for the sake of simplicity we shall use only three. Residential gateways, which are peers in this scenario, implement these priorities when the resource reservation

is performed. Based on the reservation and on the actual resource usage each gateway can decide whether it can support a new video traffic flow of a given priority.

For each video stripe, a separate multicast tree is created with the video server source as the root. However, peers are not uniformly distributed as interior nodes to these multicast trees. Instead, each peer decides, *based on its reserved available resources*, whether it can support any children for a certain stripe and, if so, it will join the network as a candidate interior node for the multicast tree corresponding to that stripe.

Using this P2P mechanism, a given node can discover potential parents for each multicast tree (stripe). When a parent is discovered, the parent will try to establish a SIP session with the originator of the message and if it succeeds the new child will receive the video stripe with a QoS guaranteed by the NGN (in case of a non-NGN scenario, the proposal is still valid since prioritisation would still be performed in the different nodes of the trees, although it would not be guaranteed in the other nodes of the transport network).

B. Peer-to-peer Architecture

Unlike other video streaming protocols using multi path delivery (e.g. SplitStream [3]), in FLaCoSt each peer having available resources can become an interior node for more than one multicast tree. To make this possible it will use *several node identifiers* to advertise its presence in the P2P overlay. Eventually, when a peer has no available uplink resources (or the initial resources have been exhausted) the peer should advertise itself using only a corresponding leaf identifier, indicating that although it is part of the network it cannot accept any children. This new concept is called *peer multiple identification* and the decision on which identifiers are advertised is taken entirely by the peer based on its wish of the peer to become an interior node for a certain multicast tree. The collection of all the peers that advertise themselves as *interior nodes* for a multicast tree is called a *peer group*. As we are using just three possible stipes, there are four peer groups: one for each multicast tree and one for leaf peers.

The rationale behind the multi-identification in the same P2P overlay is that the residential gateway peers may have reserved resources to contribute as interior nodes to several multicast trees. Hence, a peer no longer has a single multicast tree where it should always be an interior node, but rather the membership decision is dynamically taken. One of the key ideas in FLaCoSt is: every peer checks whenever the reserved and used resources change and it decides if it should advertise itself as a candidate interior node; thus belonging to one or more peer groups.

Each peer group will receive an equal portion of the hash space, with the first two bits of the hash identifier being reserved for the group. The number of identifiers each peer has is equal to the number of peer groups. These identifiers

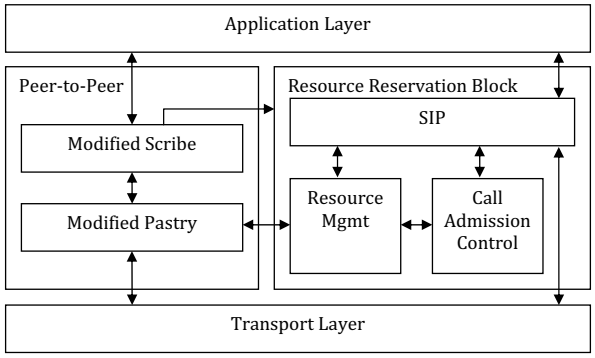

Figure 1. The protocol stack.

are unique in the overlay and are fixed during the lifetime of the peer. If a peer decides to become a member to a certain peer group, it should advertise (*positive advertisement*) to its neighbours the identifier for that particular group. Similarly, when a peer decides not to be a member of some group it will no longer reply to queries targeting the identifier of that group and in addition it will also be able to advertise the removal of the identifier (*negative advertisement*).

The behaviour resembles to having four independent overlays, where peers join and leave independently from each other. However, the advantage of using a single overlay divided in four groups is that peers in one group can help to locate peers in other groups. As long as their identifiers are advertised, routing information is exchanged between them and it is possible for a peer that joined a certain group to locate nodes in any other group. This behaviour is desirable because it enables a fast joining to any multicast tree.

With the exception of the leaf peers, when a peer becomes a member of a peer group it must also join the corresponding multicast tree to be able to fulfil its role as an interior node. If the peer cannot join the corresponding multicast tree in a finite amount of time, it is required to leave the peer group. On the other hand, peers from any peer group can join any multicast tree. The only condition is that membership to at least one peer group is required because the peer needs to have at least one identity in the overlay.

The P2P multicast protocol requires several changes to Pastry in order to support the multiple node identifiers and negative identifier advertisements proposals; and in Scribe to support tree joining only with interior nodes and unsuccessful query returns. The protocol stack (Figure 1) of each peer uses the mScribe (modified Scribe) on top of mPastry (modified Pastry). The mPastry layer is controlled by the *Resource Management*, which uses the Call Admission Control (CAC) to manage the assigned resources, reserves resources requested by SIP and approves the resource usage ([6]).

C. Modified Pastry

The mPastry protocol supports multiple identifiers for each peer, as well as re-advertisements of these identifiers to the neighbouring peers. Thus, it must also feature a slightly changed API provided to the upper layer.

1) Peer Data Structures: When each peer is created, it is assigned four identifiers, one for each peer group. The identifiers are calculated using a random common suffix part of 126 bits and a two bit prefix, taking all four possible values for the four peer groups. The node identifiers are generated by calling a Pastry initialisation function, *mPastryInit*. By default, they are not advertised to the network which means that when Pastry is initialised, no message exchanges are made with neighbouring peers. The *Resource Management* (Fig. 1) manages the membership of the peer to each peer group and, respectively, which node identifiers are advertised and which are not. This is done by evaluating the reserved resources, and it can decide, based on the requirements of the video traffic, if it can afford to accept new children in each priority class.

2) Routing: The routing process starts by selecting the leaf set and the rows of the routing table that will be used. Because it is not possible to have two local identifiers with the same two most significant digits, the algorithm selects the local identifier that shares the longest prefix with the key. In the situation when no identifiers share a common prefix, it selects the identifier closest to the key, since in this particular circumstance, the selection of a local identifier will only determine the selection of the leaf set (the routing table has a single row for these keys). Then, the algorithm checks whether the target key falls in the range of the selected leaf set. If true, the next hop is returned as the peer with the smallest distance to the key. Otherwise, the algorithm uses the routing table to determine the next peer. At this point, the only difference from the original Pastry is the selection of the row in the routing table (the selection of the column is made in the same way).

D. Modified Scribe

mScribe is intended to work with the mPastry and to manage the multicast trees. The multicast tree will still provide best effort delivery and the failure of interior nodes will be managed by the orphan peers by attempting to rejoin the multicast group. mScribe uses the following message types: JOIN, CREATE, LEAVE and MULTICAST. In the mPastry API, the *mPastryForward* implementation changes the routing behaviour of mPastry when mScribe sends a JOIN message. If left undisturbed, mPastry would route the message to the root of the multicast tree and deliver the message there. However, if the prefixes list of an intermediate peer contains the prefix of the multicast group it means that the intermediate node is a candidate intermediate node. If it is also a member of the group and if resource reservation to adopt a new child is successful, the initiator (not the sender) of the message is added to the *children* list and the mPastry routing is stopped by setting *nextID* to *null*. If the peer was not an interior node but the destination ID of the message was one of its non-advertised identifiers (meaning that the sender was not aware of an identification advertisement change), it will not prevent forward routing of the message but it will send a notification to the sender informing it that the identifier used is no longer advertised.

The *mPastryDeliver* function processes the delivered mScribe messages. The JOIN messages are handled in a way similar to the mPastry forwarding function, with the exception that the current node is an interior node for the multicast tree but did not join yet or resource reservation failed, a rejection message is sent back to the last sender. When a CREATE message is delivered, if the peer did not join the multicast tree yet and if it can be an interior node, it will join the tree and will be able to accept additional children. LEAVE messages are always directed to a particular node and will inform that node that a child has gracefully left the multicast tree. If the number of children reaches zero, and local membership for that multicast tree is no longer needed, the peer can then leave the multicast tree by sending a LEAVE message to its parent. Finally, the MULTICAST message that is usually routed to the root of a multicast tree will be forwarded to all the children in the multicast group.

E. Multicast Tree Operations

The multicast video service infrastructure is created by the video streaming servers of the provider. They use *mPastryInit* and *mScribeCreate* functions to initialise and set themselves as the root of each multicast tree. The service information (group IDs, server addresses) is published by the provider in an off-band channel.

After initialisation, the residential gateways will connect to the service using *mScribeJoin* resulting in an mPastry search in the P2P network for an interior node in the desired tree. The search stops at the first node that already joined the same multicast tree and it can accept the new peer as a child. Since by requirement these nodes are also members of the peer group associated to that multicast tree, the search is very fast because it only requires locating any peer in that particular peer group. Thus, the Pastry search is reduced to the match of a two bit prefix in the identifier. Because peers advertising their identifiers in that range are candidate interior nodes for the multicast tree, they should in general accept the new peer as child.

Eventually, it is possible that the request is sent to destination peers that are not suitable parents for the target multicast tree. This happens for example when the destination is the only peer known by the sender closest to the peer group, or when it has recently left the peer group but the negative advertisement information did not reach the sender. Under these circumstances, the destination will simply forward the

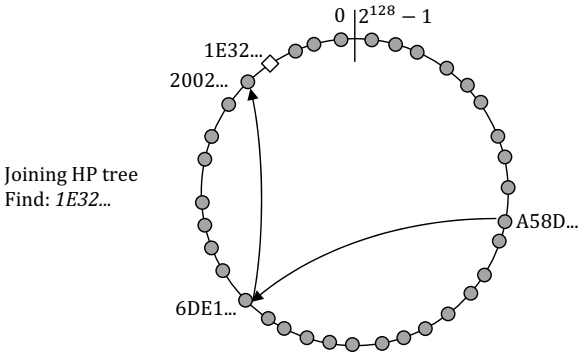

Figure 2. Joining a multicast tree through a passive intermediate peer.

query. Finally, when no path to a peer group member is found the message is returned to the last sender or to the initiator that may start looking for an alternate path.

Figure 2 illustrates a most likely join process: a new peer, member of the LP (Low Priority) peer group (and an interior node for the LP multicast tree) with node ID prefix A58D wants to join as leaf to the HP multicast tree. The root of the HP tree has the identifier prefix 1E32. To join, the new peer calls the *mScribeJoin* function which in turn will send a JOIN message to the mPastry peer closest to the group ID. Because this peer is not a member of the HP peer group (its first digit is 6 outside of the range 0 - 3), it will forward the message to its neighbour closest to the group ID. In the example, the second peer is a member of the HP peer group and will accept the initiator as child.

In FLaCoSt, peers communicate to each other to inform themselves about exceptional situation. For example, when a peer receives a message targeting one of its identifiers that is no longer advertised it will use *mPastryRemove* to inform the sender about the change, which in turn will update its routing table. To prevent a dead end when a request cannot be forwarded, a peer will call *mPastryReject* to send the message back to the sender, which is forced to look for an alternate path.

F. Resource Reservation Architecture

When a procedure in the P2P block invokes the *sipReserve* function, and prior to receiving the associated video stripe, a multimedia session is established between the peer and its recently discovered parent in the tree. The session establishment process provides the following essential functionalities to properly deliver the video stripe from the parent to the child: (1) it provides the child peer with an indication about the IP address and port where the parent will receive the RTCP reports, in case that RTP is used. Furthermore, it provides the parent peer with the IP address and port where the video will be received by the child peer, and (2) in case that the connectivity is provided to the peers by means of

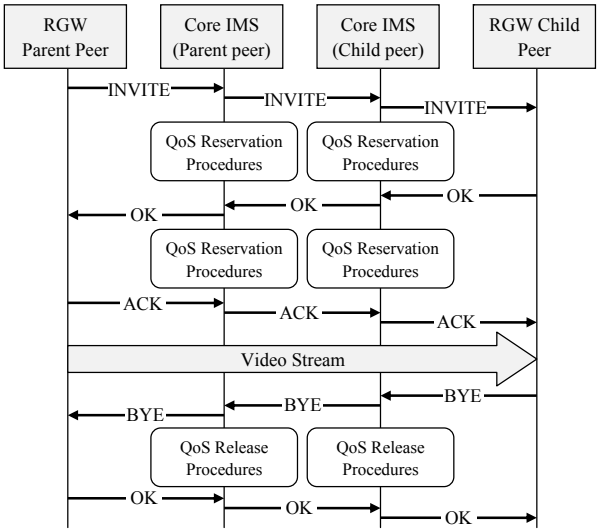

Figure 3. Session establishment and release scenario.

an NGN infrastructure, establishing the session guarantees that the service will be delivered to every peer in the tree with real end-to-end QoS (this holds as long as every peer that joins the tree establishes the multimedia session).

Figure 3 is an example of a SIP session establishment between two peers. In the example, it is supposed that each peer gets connectivity by means of a fixed broadband access line to a TISPAN NGN (as it can be seen in [7], several signalling scenarios are possible, depending on the IP connectivity access network that is being used).

The *Resource Management* block uses information provided by the CAC module and the bandwidth necessary for each stripe, to detect if there are enough resources for a given multicast tree. It exports an interface to the *mPastry* block in order to return a boolean value depending on the given priority: true if resources for that priority are available and false if there are not.

III. Experimental Validation

In this section, we present some experimental results with the goal to assess the performance and the feasibility of the multicast protocol in some usual video streaming scenarios. The results were obtained with a prototype implementation of FLaCoSt. The protocol was implemented in Java and the PeerfactSim [8] simulation engine was used to emulate the network (this simulator is not capable to generate video traffic, so the video performance was obtained with a small-world implementation).

The evaluation scenarios considered the practical usage aspects of FLaCoSt and the limitations of the simulator. Thus, we focused on a small-to-medium overlay network having from 100 to 10000 multicast-enabled residential

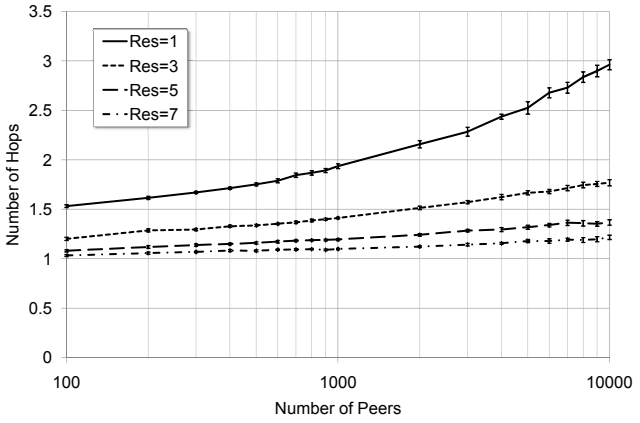

Figure 4. Average number of hops vs number of nodes.

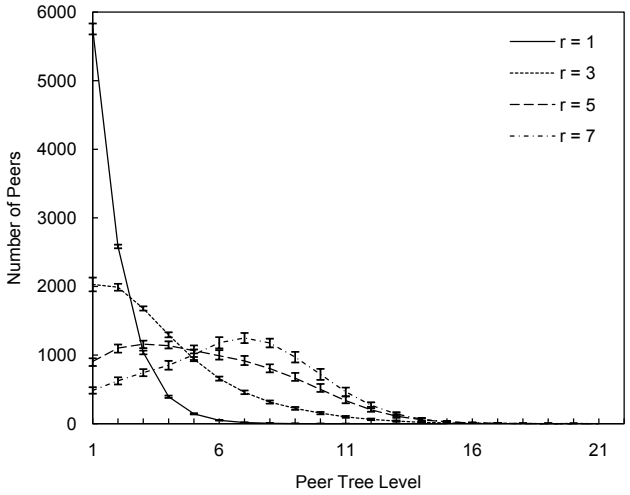

Figure 5. Number of peers at a certain level.

gateways for simulation and 25 nodes for the real implementation, with several initial resources configurations both. The mPastry layer used $b = 4$ and $|L| = 16$, and we disabled the Pastry neighbourhood set. We uniformly assigned peers to one of the eight possible classes: first class has, initially, equal res resources for all priorities (res, res, res), second class only for HP and MP $(res, res, 0)$, third class for HP and LP $(res, 0, res)$ and so on, with the eighth class having zero resources $(0, 0, 0)$. All simulations were run in five different scenarios, corresponding to four different values for the number of resources: $res \in \{1, 3, 5, 7\}$. The implementation does not include a maintenance operation; therefore the routing state converges solely from negative advertisements.

A. Routing Performance

One of the objectives of this experiment was to evaluate the performance of joining the multicast tree considering the dynamic nature of the overlay. Peers send negative advertisement whenever local resources for a multicast tree are exhausted which in turn may increase the routing distance and in the most unfavourable situations may cause peers to be rejected. Figure 4 illustrates the average number of hops versus the overlay network size for the four selected resource values.

These results validate the routing algorithm and indicate that a new node can join a multicast tree in a reasonable number of hops: negative advertisements decrease the number of hops necessary to join a multicast tree, because they reduce the number of advertised identifiers. These results were obtained using the simulator, although the small-world implementation follows these results also.

B. Peer Distribution in the Multicast Tree

One important issue to be solved when working with Application Level Multicast trees is the balance of nodes among trees. Figure 5 shows the distribution of peers for each tree level for different scenarios, changing the number of resources ($res = 1, 3, 5$ and 7). One important result from this test is that the number of peers directly connected to the root (peers at the first level) decreases when peers with a greater number of resources are introduced in the system. When $res = 1$ a given peer can have either one uplink resource or none with equal probability, thus, immediately as such a peer accepts a child it has to leave the peer group. This issue can be mitigated introducing a maintenance stage in order to balance trees.

C. Video Performance

As stated before, the simulator does not allow us to test the video performance so we used a small-world implementation, emulated using two physical PCs and the network emulator Modelnet[1] on top of them, emulating 25 user equipments. We used the Inet topology generator from the University of Michigan to generate an Autonomous System synthetic topology.

Figure 6 presents the result of the number of played streams for a given user, in a scenario where $res = 3$. This particular peer was selected among others because its position on all possible trees was the worst one: it was at level 4 in the LP tree and at the first level in HP and MP trees, so its depth deviation is high. The figure shows that the 51.72% of the time, the user is watching the video with a good quality (2 streams) while the 47.95% of the time the user gets the best quality (3 streams). Just the 0.33% of the time just 1 stream is played. Of course, these are just objective results and we do not take into account subjective results.

Figure 7 presents the average number of played streams depending on the selected *play out time*. Obviously, the

[1]https://modelnet.sysnet.ucsd.edu/

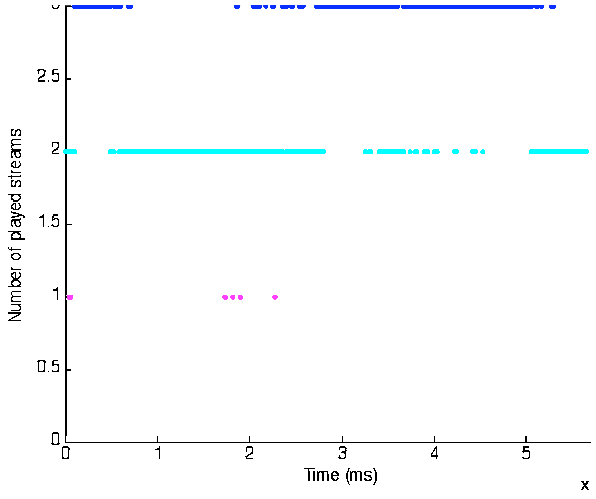

Figure 6. Played streams in time for res=3

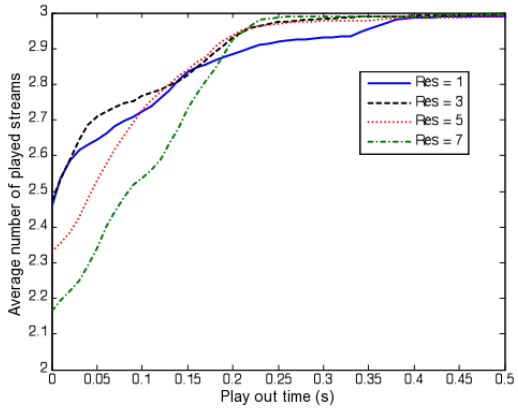

Figure 7. Average played streams for res=1,3,5,7

greater the initial buffer, the better the perceived video quality. An interesting result is that, in average, results with $res = 5$ and $res = 7$ present worst results than the others scenarios. This result is because trees are not well balanced due to the random selection of parents. Another interesting result is that the necessary play out time to achieve a very good quality is around $200ms$ for all scenarios.

IV. Conclusions

This article presents a novel P2P architecture to distribute video using multiple multicast trees. This mechanism is especially useful when the video is coded using Multi-Description Coding or Scalable Video Coding techniques, which generate several stripes: the more stripes received, the better the quality obtained. Although the proposed architecture just requires a small set of functionalities for a given node (local admission control), it increases its performance when placed in a Next Generation Network, where all devices in the whole path between peers, reserve resources for the video traffic exchange. This architecture is based on a previous work [6] where a complete RGW architecture was presented.

The P2Ps algorithms are based on well known proposals like Pastry and Scribe, modifying some of their functions in order to split the hash space in several groups. When a node advertises in the P2P network a given identifier it means that this node can be an interior node for the corresponding multicast group. This solution was validated with both simulations and a small-world emulation, presenting very good results that will be enhanced in a future work, adding maintenance stages to balance trees and to decrease the depth deviation.

Acknowledgements

This article has been partially granted by the Spanish Ministry of Science and Innovation through the CONPARTE project (TEC2007-67966-C03-03/TCM), and the Madrid Community through the BIOGRIDNET project (S-0505/TIC-0101). The authors want to thank Jose Ramon Cabezuelo for his work with the small-world testbed.

References

[1] M. Castro, P. Druschel, A. Kermarrec, and A. Rowstron, "Scribe: a large-scale and decentralized application-level multicast infrastructure," *Selected Areas in Communications, IEEE Journal on*, vol. 20, no. 8, pp. 1489–1499, 2002.

[2] M. Brogle, D. Milic, and T. Braun, "QoS Enabled Multicast for Structured P2P Networks," *Consumer Communications and Networking Conference, 2007. CCNC 2007. 2007 4th IEEE*, pp. 991–995, 2007.

[3] M. Castro, P. Druschel, A. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "SplitStream: high-bandwidth multicast in cooperative environments," *Proceedings of the nineteenth ACM symposium on Operating systems principles*, pp. 298–313, 2003.

[4] A. Nicolosi and S. Annapureddy, "P2PC AST: A peer-to-peer multicast scheme for streaming data," *1st IRIS Student Workshop (ISW03). Available at: http://www. cs. nyu. edu/nicolosi/P2PCast. ps*, 2003.

[5] A. Bikfalvi, J. Garcia-Reinoso, I. Vidal, and F. Valera, "Nozzilla: A Peer-to-Peer IPTV Distribution Service for an IMS-based NGN," in *Fifth International Conference on Networking and Services (ICNS)*, April 2009.

[6] J. Garcia, F. Valera, I. Vidal, and A. Azcorra, "A broadcasting enabled residential gateway for next generation networks," in *2nd IEEE International Workshop on Broadband Convergence Networks (BcN 2007)*, Munich, Germany, May 2007.

[7] 3GPP, "3GPP TR 24.930 V8.0.0: Signalling flows for the session setup in the IP Multimedia core network Subsystem (IMS) based on Session Initiation Protocol (SIP) and Session Description Protocol (SDP); Release 8)," March 2008.

[8] "PeerfactSim.KOM: A Simulator for Large-Scale Peer-to-Peer Networks," http://www.peerfactsim.com.