Lcast: Software-Defined Inter-Domain Multicast

Florin Coras[a,*], Jordi Domingo-Pascual[a], Fabio Maino[b], Dino Farinacci[b], Albert Cabellos-Aparicio[a]

[a] *Universitat Politècnica de Catalunya, UPC BarcelonaTECH, Barcelona, Spain*
[b] *Cisco Systems, San Jose, CA, USA*

Abstract

Traditionally, efficient inter-domain data delivery may be implemented either as a network or application layer multicast service. However, while the former has seen little uptake due to prohibitive deployment costs the latter is widely used today, but often without a minimum guaranteed performance. In this paper we present Lcast, a network-layer single-source multicast framework designed to merge the robustness and efficiency of IP multicast with the configurability and low deployment cost of application-layer overlays. The architecture involves no end-host changes and only requires the upgrading of a small set of routers to support the Locator/ID Separation Protocol (LISP), an incrementally deployable enhancement to the current global routing infrastructure. Content distribution over the Internet's core is done by means of a router overlay while within domains, end-hosts interface with Lcast using conventional multicast protocols. The overlay's scalability and topological configurability is sustained by logically centralizing group management. We illustrate the versatility of our solution by designing and assessing the scalability and performance of three management strategies for low latency content distribution. Our analysis is based on large scale simulations supported by realistic user behavior and Internet-like network topologies. The results show Lcast's low management overhead and ability to optimize delivery to meet various operational constraints. Notably, we find that it can deliver traffic with latencies close to unicast ones, independent of overlay size.

Keywords: inter-domain multicast; Locator/ID Separation Protocol; software-defined networking

1. Introduction

The Internet is gradually becoming the preferred infrastructure for delivering live content such as sports events or news to large user sets. According to recent

*Corresponding author

Email addresses: fcoras@ac.upc.edu (Florin Coras), jordi.domingo@ac.upc.edu (Jordi Domingo-Pascual), fmaino@cisco.com (Fabio Maino), dino@cisco.com (Dino Farinacci), acabello@ac.upc.edu (Albert Cabellos-Aparicio)

reports, video streaming is among the largest and the fastest growing bandwidth consumers [1] and IPTV driven revenues are to rise from less less than USD 9.7B in 2011 to USD 21.3B in 2017 [2].

For such scenarios, where one-to-many content delivery to large number of receivers is required, IP multicast [3, 4] is perhaps the most efficient solution in terms of bandwidth consumption. However, although often supported within the confinements of campus, enterprise or service provider networks, IP multicast deployments have been generally done disregarding inter-domain connectivity, thereby resulting in disconnected multicast *islands* [5, 6, 7]. One fundamental cause for the slowly advancing deployment is the requirement that *all* routers be upgraded to support the protocol. But other, frequently cited, reasons regard management complexity and the lack of a clear commercial service [8]. While the former incurs high operational expenditure the latter leads to situations when multicast implementation over links with unicast economical agreements result in loss of revenue.

The low uptake of IP multicast has led over the last decade to the development of many application-layer multicast (ALM) solutions that build end-host overlays to ensure Internet wide content dissemination. They are designed to be very flexible in accommodating globally spread users and adaptable to changing network conditions. However, the incongruence they introduce between overlay and underlying network topology diminishes their delivery efficiency with respect to that of IP multicast. Furthermore, performance analysis has uncovered significant limitations of these architectures in scaling user quality of experience with the increase of client population [9]. Reasons often reported are unavailability of inter-peer bandwidth, churn or, in some cases, insufficient upload capacity. Since they are related to either end-host behavior or upload abilities, these limitations are intrinsic to the overlay's design and, therefore, not avoidable through optimized overlay management.

In this paper, we propose a network-layer single-source multicast framework designed to merge the benefits of IP multicast and ALM while avoiding their respective deployment and scaling issues. Our goal is to enable large scale single-source streaming by interconnecting existing multicast capable domains devoid of end-host software upgrades and transparently for the greater part of existing multicast routers. This approach is complementary to existing ALM solutions as it aims to offer overlay management control to network operators in exchange for improved reliability and more efficient network resource use but at the cost of minimal infrastructure support. To achieve our goal, we exploit a unique window of opportunity offered by the development and deployment of the Locator/ID Separation Protocol (LISP) [10], a protocol meant to improve the current routing infrastructure, which we use as support for our proposal. In light of the intrinsic dependency on LISP, we refer to our solution as *LISP-casting* or shortly, *Lcast*.

Lcast creates and optimizes a LISP router overlay and transparently interfaces with end-hosts and legacy multicast routers by means of existing IP multicast protocols [3, 4, 11]. Group management functions are *logically centralized* and performed by an overlay coordinator whereby members require no prior

configuration nor need to be manually managed. We stress this as a crucial property since it circumvents the management complexity issue that plagues traditional IP multicast deployment and further opens the possibility to dynamically optimize delivery with respect to overlay topology maps. This ability could be exploited by content providers to define their own overlay coordinating algorithms or perform on-line switching between multiple ones, according to specific operational requirements or economical agreements. In this sense, Lcast is akin to Software-Defined Networking (SDN) [12] and, in fact, the two share many of the properties that derive from the implementation of a programmable control plane.

From an architectural standpoint, Lcast's ability to accommodate large number of clients is ensured through design, by decoupling the control and forwarding functions within the overlay. As a result, each can evolve, be optimized and scale according to specific needs. In particular, data plane scalability can be achieved by constraining router replication factors, to avoid performance penalties due to unicast replication inefficiency, while control plane scalability may be ensured by limiting communication overhead. In both cases the tradeoff is overlay efficiency. To assess control plane scalability, the impact of replication factors on efficiency and overlay configurability, we evaluate Lcast's ability to deliver low latency content in three distinct operational setups. Our simulations make use of (i) an Internet-like autonomous system (AS) level topology and (ii) large client traces that emulate realistic client behavior, consisting of $3k$ ASes and approximately $140k$ unique IPs obtained through a globally distributed capture of SopCast [13] overlays.

First of all, the results show the control plane's ability to scale. Even when active topology discovery mechanisms are used and client churn is high, the load is manageable by a single server acting as overlay coordinator. Second, replication factors need not be large for efficient content delivery. Finally, the overlay can be easily optimized considering various operational constraints. Notably, if inter-member latency can be estimated, Lcast can deliver content at close to unicast latencies, independent of the overlay's size.

The rest of the paper is structured as follows. We discuss the related work in Section 2 and provide a short overview of LISP in Section 3. Section 4 describes the Lcast framework and Section 5 introduces an optimization algorithm and two ways to obtain topology maps that may be used to optimize the overlay. Section 6 presents our evaluation methodology and in Section 7 we discuss the results. Finally, we conclude the paper in Section 8.

2. Related Work

As a long standing academic and commercial research challenge, single-source live media streaming benefits from copious amounts of related literature. Consequently, we restrict the discussion to a limited set of solutions and mainly focus on those that bear similarities to Lcast.

Multicast functionality that enables live media streaming was originally offered as a network-layer service. But in light of IP multicast's lack of inter-

domain deployment, network layer solutions have turned into architectures that leverage isolated multicast deployments. One notable example is MBone [14], a virtual network designed to connect multicast islands by means of static unicast tunnels. Although it was first to support distribution of content to users spread in multiple domains, it proved hard to extend since setting up tunnels involved manual configuration. AMT [15] circumvents this limitation by providing mechanisms for automatizing the tunnel setup process with the help of dedicated servers (relays and gateways) placed in source and destination domains. However, AMT does not support dynamic reconfiguration of the tunnels, i.e., of the inter-domain distribution topology. Therefore, unlike Lcast, it is not able to adapt to changing network conditions, client churn or to limit replication overhead.

A large set of application layer solutions, including NICE [16], Narada [17], OMNI [18], ZIGZAG [19] and Scribe [20], have been proposed by academia in the last decade. Out of them, OMNI [18] is the closest in spirit to our proposal. It requires service providers to deploy a set of proxying nodes that self organize in an overlay and forward traffic to subscribed clients. The optimizing algorithm employed is a distributed instance of the one we use and the metric considered is latency. In contrast, Lcast works at network-layer and is supposed to be deployed, at no additional cost, together with LISP. Moreover, Lcast's logically centralized control plane allows easy deployment of new optimization algorithms without requiring router changes.

Apart from the academic solutions, a large array of commercial ALM architectures like SopCast [13], PPLive [21], CoolStreaming [22] or UUSee [23] are widely used for Internet content streaming. Being closed source their architectures are not completely understood, nevertheless their performance has often been the subject [24, 25, 26, 27] of academic scrutiny. The results have shown significant limitations of these architectures in scaling user quality of experience with the increase of client population. Lcast, being an extension of LISP, operates on domain border routers and thus builds an overlay topology that is not directly exposed to client churn. Furthermore, through design it avoids imposing bandwidth strain on overlay members and could assure certain performance bounds.

Another approach to delivering inter-domain multicast is to connect islands of multicast enabled end-hosts by means of application layer overlays. Two of the solutions to follow this design guideline are Universal Multicast [6] and Island Multicast [7]. They are similar to Lcast in their use of existing multicast deployments for intra-island content delivery and of tunnels to connect multicast islands. However, they ensure inter-island multicast delivery by building and optimizing overlays consisting of end-hosts. This gives rise to three fundamental differences. First, Lcast does not require changes to end-hosts as the inter-domain router overlay seamlessly interfaces with local multicast. Second, Lcast should use more efficiently the underlying intra-domain network since packet replication is always performed in domain border routers and therefore packets avoid traveling intra-domain prior to being replicated and forwarded to hosts in foreign domains. As a downside to this, when fan-out values are large, the

routers have a higher processing overhead however, as shown by our experiments, they need not be large for good performance. Finally, all solutions offer the option to constrain fan-out values but Lcast offers control to operators who have a vested interest in the efficiency of the overlay. That is, router out degrees should be generally limited to protect routers from saturating their interfaces and to ensure fairness in distributing the replication responsibilities. We then believe that providing the ability to configure fan-out actually makes Lcast better suited for operational deployments than its P2P counterparts.

We previously proposed CoreCast [28], a LISP inspired inter-domain streaming architecture where source and client routers operate according to a client-server model. Both CoreCast and Lcast are based on LISP protocol mechanisms but they have quite different approaches to delivering inter-domain traffic. In this sense, CoreCast aims to diminish inter-domain bandwidth use when compared to P2P live streaming systems while Lcast aims to provide a scalable and easily reconfigurable offloading mechanism for the source LISP router. Finally, it is worth mentioning that LISP-Multicast [29] integrates IP multicast functionality into LISP. Naturally, it inherits all the properties of traditional network layer multicast however it also requires core router support for inter-domain use, thus making it unfeasible for a wide-scale deployment.

3. LISP Background

LISP [10] is an architectural solution to the Internet's scalability problem, recently generated by the alarming growth of inter-domain routing tables [30]. To this end, LISP's main goal is to split the semantics of IP addresses with the aim of forming two namespaces that unambiguously identify core (locators) and edge (identifiers) network objects. Its development is aided by a pilot-network [31] and a considerable community spanning over 32 countries, with members pertaining to both academia and industry. Notably, the protocol that has recently undergone IETF standardization [32].

One of LISP's defining features, that also distinguishes it from many of its competitors, is its support for an incremental deployment. It is transparent to end-hosts and requires the upgrading of only stub domain border routers. Moreover, to facilitate transition from the current Internet infrastructure, both locator and identifier namespaces use the existing IP addressing scheme. Therefore, the split does not affect routing within existing stub or transit networks. Nevertheless, as identifiers and locators bear relevance only within their respective namespaces, a form of conversion, from one to the other, has to be performed at border points between core and edge networks. LISP enabled border routers make use of a technique called map-and-encap [33] for the translation.

Apart from the need for data plane modifications, map-and-encap also requires the introduction of a new control plane *mapping function* able to provide bindings that link identifiers to locators. Therefore, prior to forwarding a host generated packet (see Figure 1), a LISP router maps the destination address, or what LISP calls an *endpoint identifier* (EID), to one or more corresponding destination *routing locators*(s) (RLOC) by means of a *mapping* obtained from a

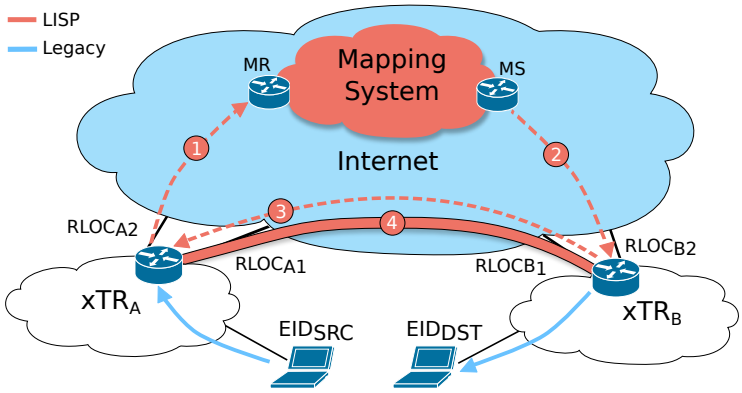


Figure 1: Example packet exchange between EID_{SRC} and EID_{DST} with LISP. Packets travel from EID_{SRC} to xTR_A according to intra-domain routing. xTR_A obtains a mapping binding EID_{DST} to $RLOC_{B1}$ and $RLOC_{B2}$ from xTR_C through the mapping-system (steps 1-3). Then, xTR_A chooses $RLOC_{B1}$ as tunnel destination, encapsulates the packets and forwards them over the Internet's core to xTR_B (step 4). Finally, xTR_B decapsulates and forwards the packets to EID_{DST}.

LISP specific directory service, called the *mapping database system* [34, 35]. A *priority* is associated to an RLOC to indicate the preference of it being used, and a *weight* indicates how traffic is to be load balanced between RLOCs with the same priority. Mappings are requested on-demand, as opposed to being proactively obtained, and stored in a local *map-cache* such that they may be reused. To retrieve a mapping, a LISP router directs a `Map-Request` message carrying the destination EID to a *Map-Resolver* (MR) part of the mapping system and in return receives as answer a `Map-Reply`. Once the mapping is obtained, the border router selects one of the EID's RLOCs as tunnel destination, encapsulates the packet with a LISP-UDP-IP header and forwards it to corresponding edge network. At the receiving router, the packet is decapsulated and forwarded to its intended destination. For a detailed example see Figure 1.

In LISP parlance, the source router, that performs the encapsulation, is called an Ingress Tunnel Router (ITR) whereas the one performing the decapsulation is named the Egress Tunnel Router (ETR). One that performs both functions is referred to as an xTR. Additionally, LISP makes use of Re-encapsulating Tunnel Routers (RTRs), that perform re-encapsulation, i.e., decapsulation followed by encapsulation, to enable packet re-routing based on EID.

It is also in the ETR's duty to register the EID address space, one or multiple EID-prefixes it is responsible for, with its associated Map-Server (MS) by means of `Map-Register` messages. At their turn, Map-Servers aggregate and advertise the EID-prefixes in the mapping system to enable EID routing. Then, having this information, the mapping system can ensure the delivery of a `Map-Request` to the MS that originates the EID-prefix covering the request's destination ad-

dress. Depending on configuration, the MS can either answer or forward to an ETR the `Map-Requests` it receives.

4. Lcast Architecture

This section presents our proposal for supporting inter-domain multicast streaming. We start by providing an overview of the architecture and then describe in greater detail group management procedures and signaling.

4.1. Architecture Overview

Lcast is a LISP extension that provides a single-source multicast service to clients in disjoint multicast islands by means of a router overlay. It compensates for the lack of an inter-domain multicast infrastructure by performing unicast encapsulated, and if possible also multicast encapsulated, replication of multicast traffic across the Internet's core. The resulting overlay interfaces with existing intra-domain IP multicast protocols so it does not require any end-host software upgrades. All member domains must be LISP enabled and may participate in the overlay with at least one of their border routers (ETRs). On the data path, the source domain's border router, an ITR, heads the distribution tree and is the first to perform encapsulated replication. Subsequently, all downstream overlay members, save for the leaves, replicate the received packets up to a certain fan-out. Note that since traffic is unidirectional, from multicast source to clients, all routers but the source ITR perform either only decapsulation or decapsulation and re-encapsulation. For brevity we refer to all of them as ETRs, although those that perform both functions also implement RTR functionality. See Figure 2 for a depiction of an example Lcast data plane.

An important drawback to unicast encapsulated replication is that it reduces throughput proportionally to the replication factor, if performed multiple times out the same interface. As a result, increasing fan-out can quickly saturate router interfaces and therefore not only deteriorate overlay performance but also congest other flows sharing the same links. Additionally, since packet replication is performed sequentially, the time difference between the instance the first and last replicas are forwarded may be considerable. So, besides increasing the delay to obtaining the multicast packets for directly connected downstream members, the resulting latency may accumulate and distribute unevenly across the hierarchy, randomly leading to branches with low performance. Finally, apart from the limitations concerning performance, unbounded fan-out can also lead to unfair and/or economically unfeasible situations. Generally, Lcast and other island multicast solutions substantially reduce inter-domain traffic exchange, if compared to simple unicast delivery or unoptimized P2P overlays (see Section 7). However, if the distribution tree is not carefully constructed, member routers serving few clients might be requested to replicate a disproportionate number of times, against their interest and to the advantage of other economically benefited peers. To avoid these inefficiencies and in the interest of fairness, we request that Lcast members have a *constrained fan-out* whereby the overlay

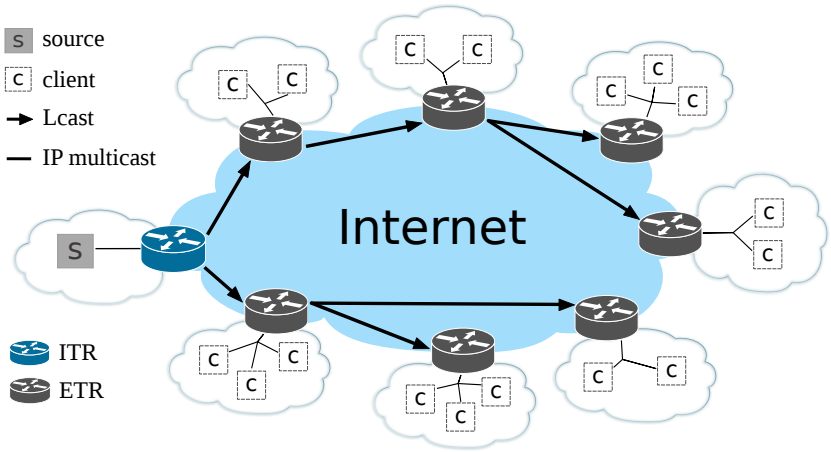


Figure 2: Example Lcast data-plane architecture. The ITR is the first to replicate the content and all downstream ETRs may replicate up to a fixed fan-out value. In the example, fan-out is constrained to 2.

must be organized as a degree-constrained tree, despite the potential to reduce distribution efficiency. Fan-out values could be fixed for the whole overlay or reported at subscription by each member.

It should be noted that, similarly to end-host overlays, purely replicating routers, RTRs, could be provisioned in transit domains to ensure improved overlay stability, performance and also considerably reduce or remove altogether the replication overhead of ETRs. However, such a solution also implies a business model, different to the one discussed here, where third party entities manage the RTRs and engage in economical agreements with the source and client edge-domains. Since we are not ready to model business relationships, or speculate how such RTRs could be deployed, in this paper we limit the analysis to overlays where replication is performed exclusively by ETRs. For a technical discussion on how RTRs could be configured to participate in the overlay, we refer the interested reader to an Internet-Draft [36] we published on the subject.

Another limitation to having routers participate in an overlay, is that they are generally inefficient at handling complex computation tasks since they are designed to perform fast forwarding of packets as opposed to general purpose computing. To circumvent this drawback, Lcast leverages LISP's native separation between data and control plane to ensure the logical centralization of group management functions in an overlay coordinator. Thereby, at data plane level routers only perform encapsulated replication while at control plane level, the coordinator must compute a distribution tree and ensure members are organized according to it. Besides supporting our original goal of having no management costs for routers, this design also opens the possibility for enhanced overlay configurability. In this sense, if the coordinator obtains or is configured with a map of the locator underlay, it may proceed to optimize the distribution tree with

respect to a given metric. Moreover, the architecture allows the switching between multiple optimization functions or metrics, even on-line, to meet changing operational requirements. Note that we are not the first to propose such split. This ideas has been previously recommended to aid routing scalability [37] and is nowadays central to current SDN research [38].

Possible implementers of the overlay coordinator may be the source ITR or the MS. Reusing the previous argument, since the ITR is a router, we consider the MS better fit for the function. In fact, due to its position in the LISP control plane, the MS is required to process and provide answers to all `Map-Requests` originated by routers willing to initiate communication with the multicast source. Then, as it can recognize and keep track of all overlay members, the MS should also be the one to decide the attachment point for a joining member or the one to optimize the overlay. Although the design allows for the overlay state and/or management functions to be distributed, in this article we are interested in evaluating if the control plane overhead is sustainable by one, off-the-shelf, server acting as MS.

Lcast is compatible with the current LISP specification, but apart from the canonical LISP messages we introduced in Section 3, it additionally requires the signaling messages defined in [39] for conveying joining (`Join-Request` message) and leaving (`Leave-Request` message) multicast information. They are not Lcast specific and have been designed to simplify the connecting of multicast capable sites with LISP-Multicast.

4.2. Member Subscription

In Lcast enabled domains, end-hosts request single-sourced multicast content, the way they do with traditional IP multicast, namely, by subscribing to a multicast stream using the Internet Group Management Protocol (IGMP) [40, 4]. They learn the *channel identifier* (S-EID,G), consisting of the multicast source address S-EID and a multicast group address G, used to distinguish between the multiple channels originated by a source, either through configuration or with the help of application-layer protocols. Except in the particular case when S-EID is part of the same domain, and therefore the content may be delivered locally without Lcast, the subscription request propagates intra-domain up to one of the domain's border routers, an ETR. If the ETR is already a member of the multicast channel, it starts replicating the multicast content towards the requesting end-host and no further action is taken. If not, the ETR initiates a two step overlay join procedure whereby it first attaches to the Lcast overlay serving (S-EID,G) and secondly it advertises its ability to replicate multicast content.

To complete the first step, the ETR must initially obtain the locator, of at least one of the already connected routers, to be used as overlay attachment point. It achieves this by requesting that the channel identifier be mapped to the locators of potential overlay parents, in essence, by sending a `Map-Request` for (S-EID,G). The request propagates through the mapping system up to the coordinating MS, which ensuing the request's receipt, starts a search for overlay

members with spare capacity. The search may be done randomly or, if additional topological information exists, in accordance to a predefined heuristic that could ensure that an optimal attachment point is chosen. Once obtained, the result, consisting of a list of one or multiple RLOCs pertaining to the overlay members able to accommodate new children, is conveyed to the joining ETR in a `Map-Reply`. Typically, the MS will offer an ETR the possibility of choosing its upstream either when not optimizing the overlay or when all the choices have an equal cost in the distribution tree. Using local policy (e.g., shortest AS path) and the priority and weight values associated to the list entries, the ETR choses the best RLOC and sends it a `Join-Request` message to request the setting up of an overlay branch between the two. The parent router appends the RLOC of the joining ETR to the list towards which it performs unicast encapsulated replication, therefore concluding the ETR's attachment. Alternatively, if the two routers can be connected by inter-domain multicast, the joining ETR first performs a protocol dependent multicast join to the parent in the underlying inter-domain network. Afterwards, it indicates the multicast channel identifier, to be used as destination for the multicast encapsulated packets (as opposed to unicast encapsulated), in the `Join-Request`.

A special case arises when an ETR is first to join the overlay. In this situation, the ETR requests the multicast content from the ITR, but it may happen that the ITR is not yet subscribed to (S-EID,G). Therefore, on receipt of the `Join-Request`, the ITR must first subscribe to the multicast source, using IGMP or a PIM Join [41] message, to obtain the streamed content to be replicated towards the joining ETR.

The second step in a member's subscription procedure is to signal that it can perform replication within the overlay. To this end, once the ETR is attached, it starts registering (S-EID,G) with the MS. The `Map-Register` message conveys the ETR's RLOC, that of the chosen parent and the number of intra-domain clients it serves at a certain time instant. To be noted that the estimating of the membership in a multicast session, although traditionally a difficult task, can be achieved within a domain using the explicit tracking capabilities of both IGPM and PIM. Then, having for a channel identifier the registration messages of all the overlay members, allows the MS to build an aggregated (S-EID,G) mapping that provides a complete view of the overlay's topology, i.e., of how the members organize in a distribution tree. This mapping, together with underlay topological information, if any exists, is used by the MS to answer `Map-Request`s of joining ETRs and to optimize the distribution tree. Two additional benefits of the procedure, since registrations are refreshed periodically, are that the MS implicitly detects the failure of an ETR and also becomes aware of the changes in client population within a domain.

The way the distribution tree is built has two advantages. First, it ensures that multicast packets in the source domain do not reach the ITR if no ETR is joined and the ITR does not participate in a local multicast group. Second, packets are forwarded from ITR to all ETRs without mapping database lookups thus, with minimum overhead.

4.3. Member Failure and Unsubscription

If a member loses network connectivity, its data path children will sense the failure either as a lack of multicast packets or by means of a LISP specific mechanism, called *RLOC-Probing*. This procedure, used by xTRs, consists in the use of `Map-Request` messages to determine reachability of peer xTRs and to estimate round-trip times (RTT). Once the children detect the failure they look for new overlay parents by either sending `Join-Requests` to other RLOCs in the mapping associated to (S-EID,G) or, if no other exists, by redoing the whole subscription procedure. Still, such circumstances will result in packet loss for all members of the subtree headed by the affected router and out of band mechanisms would be required for remedying the failure. Such mechanisms are out of the scope of the current paper. However, sudden loss of network connectivity for a domain's border router should be a seldom occurrence.

An ETR initiates its unsubscription from the Lcast overlay when the last of its clients leaves the intra-domain multicast group. First, if the ETR replicates content to other overlay members, it increments the priority of the RLOC registered with the MS to the least preferable value and replies to RLOC-Probing messages indicating that its RLOC is unreachable. The update forces the MS to avoid assigning the ETR new children and forces the existing ones to find new overlay parents. The lookup procedure is identical to the one followed in the event of a parent failure however, in this case there are no packet losses. Alternatively, when the MS senses the departure of an ETR, it could proceed to optimizing the whole affected subtree to avoid performance degradation. As a second step, the ETR sends a `Leave-Request` to its overlay parent and stops registering (S-EID,G) with the MS, concluding the unsubscription.

4.4. Distribution Tree (Re)Configuration

The position of an overlay member on the data-path is established at subscription time, however the MS could be configured to further optimize the distribution tree, if provided with information about the overlay's topology. In this case, distribution efficiency is controlled by the MS through optimal placement of joining ETRs and/or through periodic or enforced tree reshaping.

When reorganizing the distribution tree, the MS informs members of their new positions through updated mappings. To avoid packet loss and to assure a seamless transition, members use a *make before break* procedure when switching parents. Hence, prior to sending `Leave-Request` to their overlay parents, members first attach to those indicated in the updated mappings. If any duplicate packets arise, they should be discarded by end-hosts at application layer.

This type of centralized management enables the easy customization of the distribution tree as routers are oblivious to optimization algorithm changes. In fact, a *key feature* of Lcast, is that an operator in control of the MS can perform on-line switching between multiple optimization algorithms or topology discovery protocols, if more are supported, to better adapt the overlay to operational constraints. In the next section, we provide as example a possible tree optimizing algorithm and ways of obtaining topological information that could be implemented in Lcast.

5. Overlay Optimization

The configuration of the Lcast overlay controller is application and operator specific. To illustrate this point, in this section we consider the classical problem of delivering latency constrained content (e.g., live streaming of news and sports events) and show how Lcast could be used to solve it. We first propose an algorithm to compute the distribution tree and afterwards point out how ITR-local BGP routing tables and organized inter-domain latency measurements could be used to approximate overlay topology. For brevity, we further refer to the combination of an optimization algorithm and a topology discovery mechanism as an *optimization strategy*.

5.1. Distribution Tree Optimization Algorithm

In what follows, we use the term *distance* when referring to a relative length or amplitude of a metric, observed on a path connecting two points, but when the exact nature of the metric is of no interest. Considering our goal of delivering content for delay sensitive applications, the function we minimize in our experiments is the maximum distance (e.g. latency or number of AS hops) from a client to the multicast source. Notice that the reference is the end-host and not the domain border router (ETR). Thus, what matters in deciding an ETR's position in the overlay tree is not solely its distance to the ITR but also the number of clients it serves. Then, a router close to the source but serving few clients might find itself lower in the hierarchy than another with a slightly higher distance but with a larger client set. In other words, the algorithm tries to improve average end-host quality of experience by optimizing the router overlay considering two dimensions, inter-router distance and the size of the client set served by a router. This also ensures the algorithm is fair to members. Domains with fewer clients are more likely to become leaves while those with larger user sets, the ones that benefit most from Lcast, are required to contribute by replicating.

The problem described above, henceforth named *minimum average distance, degree-bounded spanning tree (MADDBST)*, may be formally stated the following way:

Definition 1. *Given an undirected complete graph G=(V,E), a designated vertex* $r \in V$*, a degree bound* $d(v) \leq d_{max}$*,* $\forall\ v \in V$*,* $d_{max} \in \mathbb{N}$*, a vertex weight function* $c(v) \in \mathbb{N}$ *and an edge weight function* $w(e) \in \mathbb{R}^+$*,* $\forall$ *edge* $e \in E$*. Let* $P^T_{r,v}$ *be the set of edges e on the path from vertex r to v in the graph's spanning tree T. Also, let* $W^T_{r,v} = \sum_{e \in P^T_{r,v}} w(e)$ *represent the cost of the path linking r and v in the spanning tree T. Find the spanning tree T of G, routed at r, satisfying* $d_T(v) \leq d_{max}$*, such that* $\sum_{v \in V, v \neq r} c(v) W^T_{r,v}$ *is minimized.*

We note that [42] and [18] have previously defined and solved similar optimization problems. Shi et al. [42] also proved that a particular instance of the problem, where all vertices have weight 1, is NP-complete for degree constraints $2 \leq d_{max} \leq |V| - 1$. Similarly to our approach, they were interested in

a centralized solution whereas Banerjee et al. [18] have successfully managed to distribute the algorithm.

The heuristic we used to solve the MADDBST problem is similar to the one used by Banerjee and it is a variant of the one proposed by Shi. In short, the algorithm works by incrementally growing a tree started at the root node r until it becomes a spanning tree. For each node v, not yet a tree member, it selects a potential parent node u in the tree T, such that the metric $\delta(v) = (W^T_{r,u} + w(u,v))/c(v)$, i.e., the distance to the source per client, is minimized. At each step, the node with the smallest metric value is added to the tree and the parent selection is redone.

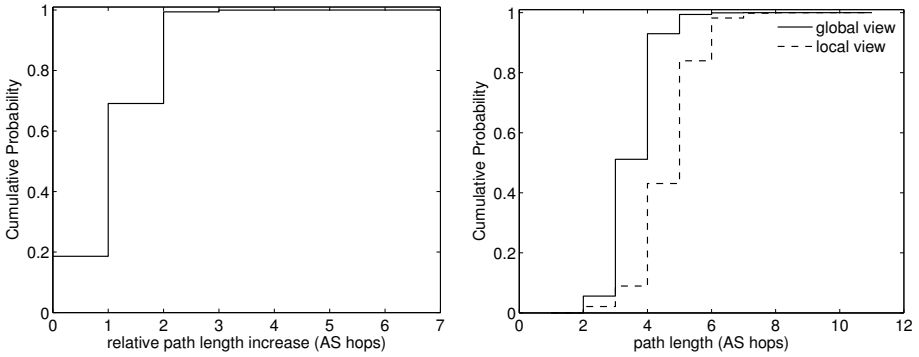
5.2. BGP-based Topology Map

One of the best sources of topological information that is not or can not be commonly used by application layer overlays is the BGP routing table. The BGP information an AS router holds attempts to present an Internet wide interconnection map. But, due to the algorithm's distributed nature and its use of policy, both inaccuracies and incomplete information may exist.

The ITR has two options for obtaining BGP topological information. First, it may aggregate partial BGP feeds from multiple overlay members (*global view*) or second, it may itself connect to BGP (*local view*). The former could ensure a more detailed description of the topology, and thus grounds for better decisions, while the latter a more restricted, partial view of the interconnection map and seemingly worse performance. Another aspect to be considered regarding the global view is that an off-line obtained BGP map may be rendered inadequate due to churn whereas one obtained through on-line aggregation of multiple BGP tables may be a technically challenging task. Even more so as some domains may be reluctant to provide such information which they often deem as sensitive. By contrast, the local, on-line topology gathering mechanism requires nothing more than BGP feeds from the ITR. Additionally, there is no need for a communication protocol between the MS and the overlay members for the conveying of BGP reachability information.

To compare the two alternatives, we take as global view the Internet-like topology we use in our evaluation and as local view the routing table of the ITR. More details on how we obtained the dataset can be found in Section 6.2. Using these two topologies we computed the relative AS path length increase of the local view and the distribution of the path lengths for both. Results are depicted in Figure 3. If we focus on Figure 3a, we see that 99% of the local view paths are at most 2 hops longer than in the global view and about 20% have an identical length. On average, path length in the local view increases only about 1.1 hops. This is also illustrated in Figure 3b where we can also note that, save for the average 1 hop increase, the distributions of hop lengths are similar. Given the relatively small difference, we are lead to conclude that the local view presents a reasonably accurate description of the topology.

Due to its relatively good accuracy and, more importantly, due to the implementation simplicity, we opted in our experiments for the BGP topology discovery mechanism based on local information. The metric it provides, inter



(a) Cumulative probability of the relative path length increase due to local view

(b) Cumulative probability of the AS path lengths

Figure 3: Comparison of BGP local and global views

AS hops, in combination with the optimization algorithm results in a degree-constrained shortest AS path tree optimization strategy. For brevity, we shall refer to it as *bgp*. Should there be interest in obtaining a minimum AS hop cost tree, at the expense of larger number of hops to the source, a degree-constrained minimum spanning tree heuristic should be used.

5.3. Latency-based Topology Map

Inter-member latency is a metric commonly employed by application layer overlays in topology optimizations. Yet, obtaining an inter-member latency map may scale poorly with the population size and therefore its implementation may be both expensive and technically challenging. For instance, in a topology consisting of N members, a naive approach, whereby each member measures all possible peers, would require N-1 measurements per member. To prevent scaling the number of measurements with the size of potentially large overlays, a more intelligent approach for the selection of link latencies worth estimating is needed.

We avoid performing a large number of measurements and assure they are carried out in an optimized order by exploiting a mechanism similar to the one used by Banerjee et al. for the group management of NICE [16]. The solution consists in clustering nodes that are close to one another in terms of latency and limiting the inter-member measurements to just the pairs finding themselves in close proximity. The amortized cost analysis shows that the number of control plane peers (i.e., the number of peers measured) at an average member is constant $O(k)$ while in the worst case it can reach $O(k \log(N))$. Where, k is a constant limiting the node degree and the size of the cluster and N is the number of overlay members. Even in the worst case, given that N may be in the range of thousands to tens of thousands, this is a considerable decrease from $O(N)$.

Another advantage of the centralized group management is that the latency discovery mechanism, when implemented in Lcast, has a lower per member communication overhead than in NICE, as members do not participate in a separate control plane protocol. However, LISP's extension to provide for a simple mechanism to convey latency measurements between ETRs and the MS is required. Since ETRs check the liveness of the locators associated to cached mappings with RLOC-probing, the extension requires just the implementation of a message, similar to a `Map-Reply` for reporting the RTT estimates.

The combination of the latency topology discovery protocol and the optimization algorithm results in an optimization strategy we further refer to as *lat*.

6. Evaluation Methodology

To compare the performance of the overlay optimization strategies proposed in the previous section, we implemented an event-based simulator. In what follows we describe the simulator's components and our evaluation methodology. We start by describing the datasets and the procedure followed to build an Internet-like inter-domain topology. Subsequently, we present the methodology used to generate traces that emulate realistic client behavior and explain our simulation setup. We conclude the section with a brief presentation of the metrics used to evaluate overlay performance.

6.1. Simulation Methodology

Our experimental evaluation simulates a set of $140k$ end-hosts spread in $3k$ autonomous systems, watching a live stream over an Internet-like topology with the help of Lcast. For this purpose, we developed a event-base simulator capable of handling large scale Lcast overlays and several optimization strategies. This resulted in the partial implementation of Map-Server and ETR functionality.

In all the experiments we employ as content source an arbitrary autonomous system as we observed that the choice does not influence the results. Client ASes participate in the overlay with one ETR and their decision to subscribe or unsubscribe is triggered by the activity of intra-domain users they serve. To simulate various types of user behavior, the latter is provided as input to the simulator in the form of trace files that log end-host join and leave events. ETR subscriptions are not optimized, but done at the first randomly found free position in the distribution tree not to bias the effect of the optimization strategies and are always based on unicast connections. The distribution tree is optimized by the MS periodically (10 min) or if more than a third of the members sustain an increase of the served client set above 10 or drop to 1, join or leave the overlay. These values were chosen to balance the computation costs and the overlay's content delivery efficiency. Additionally, to evaluate the influence of tree optimizations on communication overhead, we require that all member departures trigger the optimization of the affected sub-trees instead of only having the affected children reperform the subscription procedure. We

detail the optimization algorithm, the Internet-like topology and the three traces that describe user behavior in the next sections.

The performance of each optimization strategy is evaluated by running simulations with respect to the client trace and fan-out values, which we vary between 2 and 10 to understand how replication factors influence performance. For each such simulation run, we sample and store for analysis overlay state once per minute and control traffic overhead once per second.

Finally, to better gauge the performance of *bgp* and *lat*, we also define and evaluate a very simple overlay management strategy that does not perform topology discovery or tree optimizations. In this scenario, we further refer to as *rnd*, members join at random positions in the distribution tree and member departures always require the affected children to repeat the subscription procedure.

6.2. Internet Inter-Domain Topology

To obtain a realistic global inter-domain topology we aggregated datasets that estimate how autonomous systems interconnect from multiple sources: iPlane [43], RouteViews [44], CAIDA [45] and RIPE [46]. All the data used is from April 2011. The dataset lacks link specific BGP policy information that could transform part of the AS graph's edges in arcs (directed links). Most affected by this assumption are the links between customers and their upstream providers and peering links between stub ASes. The first type may not be used by upstream providers for transiting traffic to destinations other than those found in their clients' network. The second type may not be used to transit traffic to destinations outside a peer's network. Since, Lcast only replicates traffic between stub domain border routers, the two types of links may be misused only when a non-member stub domain transits traffic to or from an Lcast member. However, stub domains generally have a much less diverse connectivity than transit domains thereby, such situations should be a seldom occurrence.

For the resulting inter-AS topology, we observed that the log-log plot for the complementary cumulative distribution function (CCDF) of the AS-node degree follows a straight line, a property found in power law distributions. Accordingly, as previously shown in [47] and [48], the Internet AS topology is a scale-free network with power law node degree distribution. Further, the average path length in our topology is 3.5 or 5.4% lower than the 3.7 observed [49] in the Internet. These two results corroborate our claim that the aggregate topology has properties similar to those of Internet's AS graph.

For estimating inter-AS latency, we made use of iPlane's [43] proven latency prediction abilities for IP pairs [50]. Because we needed to estimate the latency between domain border routers we had to elect for all participating ASes a representant. We did so by using iPlane's estimations that associate points of presence (PoP) to ASes and their inter-connection map. For any domain, the PoP with the largest degree was elected as the representant. In about 30% of the cases, when iPlane failed to provide an answer, we used a latency estimator based on geographical distance described in [51].

6.3. The Client Traces

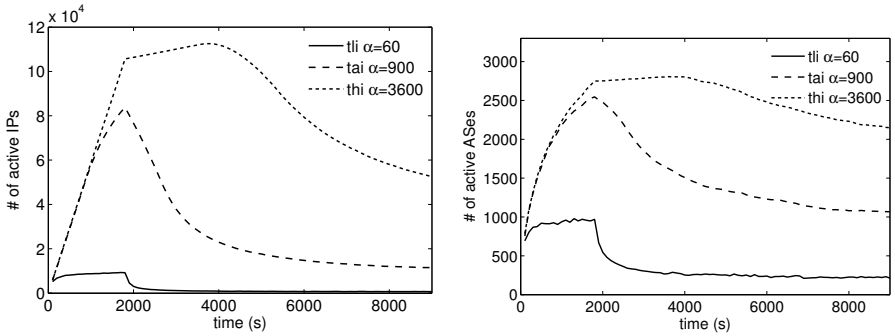
To ensure a thorough evaluation of the optimization strategies, we make use of client traces that emulate complementary types of user behavior. The domains that participate in the overlay and their respective number of clients were obtained from a passive distributed capture of several P2P TV channels whereas the client churn was modeled in accordance to recent results in the field. We detail both efforts in what follows.

SopCast [13] is one of the P2P TV applications frequently used for streaming of live sports events. Wanting to model client distribution for large events of global, or at least wide-spread interest, we captured the traffic pertaining to several SopCast overlays during an 2011 UEFA Champions League semifinal. To this end, we used 2 vantage points in USA, 5 in Europe and 2 in Asia, spanning a total of 6 countries. We were interested in understanding how clients cluster in autonomous systems, not in the specific performance of a channel's overlay. Thus, depending on the upload capabilities of each vantage point, we joined a number of P2P channels, streaming the same event, at each node. As a result, the traces finally contained more than 145k unique IPs spread in over 3.8k ASes. Out of them, in our simulations we used 3k ASes for which we could compute pairwise latency estimates. More information about the traces we captured and their properties can be found in [52].

In spite of the large size of our captured dataset, lack of logs from the overlay's bootstrapping server made it impossible to approximate client lifetime in the overlay. We thus resorted to synthetic modeling of client churn. As shown by several studies [53, 25, 24, 27, 26], it is generally acceptedthat client arrival process, at least for periods spanning dozens of minutes, can be modeled by a Poisson process. Furthermore, Sripanidkulchai et al. observed in [53], after analyzing 3 months worth of Akamai logs, that *short duration* events, which last a couple of hours, present flash crowds whereas non-stop streams have a time of day behavior. These findings were confirmed by Veloso in [27] who also noted that for *long* streams client inter-arrivals can be modeled through a Pareto or a piecewise stationary Poisson process.

For client session lengths however, consensus could not be found. Thus, depending on stream length or the type of system being analyzed by either paper, they may follow different distributions. Still, with the exception of [25], there seems to be an agreement that sessions should have lengths distributed according to a power law but opinions diverge when assessing the weight of the tail.

Considering the works discussed above, in order to perform an evaluation of our proposed architecture that acknowledges the wide range of client behavior, we generated 3 traces with complementary properties. The goal was to model a short event, spanning 2h 30min, with a piece-wise Poisson arrival process but with different shapes for the session length distributions. In order to capture the flash crowd effect we required that 80% of the clients join during the first 30min, and the rest spread over the time left. For the session lengths we used a Pareto distribution with a shape parameter of 1.5 and a scale parameter,



(a) Clients (b) ASes

Figure 4: Number of active clients and ASes for the generated traces with time.

denoted α, that took the values 1min, 15min and 1h in order to emulate low, average and respectively high client interest in the streamed content. Figure 4 depicts the evolution of the number of active clients and ASes with time for the tree traces. For brevity, we shall refer to them as *tli*, *tai* and *thi*, respectively, as a shorthand of the client interest modeled in each case.

The first trace represents the worst case scenario from overlay stability perspective, the reason being that clients leave the stream soon after joining, that is, they perform what's known as *channel surfing*. As the total number of active clients both in the overlay and within each AS is the lowest, the trace is also good for evaluating the low bound of the overlay's efficiency. Conversely, with *thi* we can assess Lcast's efficiency and ability to optimize overlays with large number of clients in low churn conditions. In light of the generation procedure, each of the three traces should be a good approximation of specific but realistic client behavior when part of a multicast group. However, if considered together, they should provide a good coverage of all practically encountered behavior. The client traces along with the SopCast ones can be found at `http://www.cba.upc.edu/lcast`.

6.4. Metrics

We evaluate the performance of the proposed schemes along the following dimensions:

- **latency stretch**: this metric measures a client's relative gain in latency to the stream's source when compared to the unicast one way delay between the source and the client. While a value lower than 1 indicates that Lcast delivers packets faster than unicast, a value larger than 1 does not necessarily imply a large absolute delay.
- **hop stretch**: it measures a client's relative gain in number of AS hops to the stream's source when compared to the number of hops on the unicast path linking the two.

- **tree cost**: is a metric we define to quantify Lcast's efficiency in using underlying network resources. It is computed as the ratio of the number of AS hops crossed for the delivery of one packet to all end-host clients to the number of AS hops crossed when using unicast for the same purpose:

$$\frac{\sum_{v \in V} hop(v, p_v)}{\sum_{v \in V} c(v)\, hop(v, root)} \tag{1}$$

  where V is the set of all member routers, $hop(\cdot)$ is a function that returns the number of AS hops between two routers, p_v is the overlay parent for v and $c(\cdot)$ is a function that returns the number of end-hosts served by a member router. Tree cost is lower than 1 if the overlay is more efficient than unicast delivery.

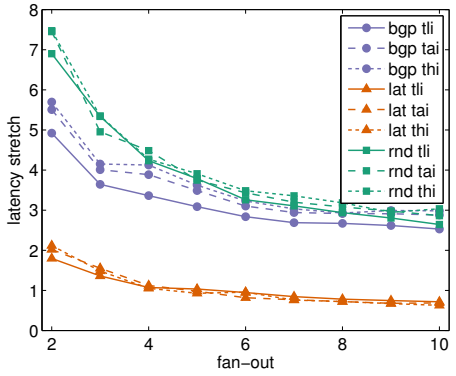
- **control traffic overhead**: to evaluate the scalability of Lcast's control plane we use a set of metrics that measure the number of messages exchanged by the MS with the tree members for the purpose of creating a tree, maintaining tree integrity, tree optimizations and topology discovery.

7. Results and Discussion

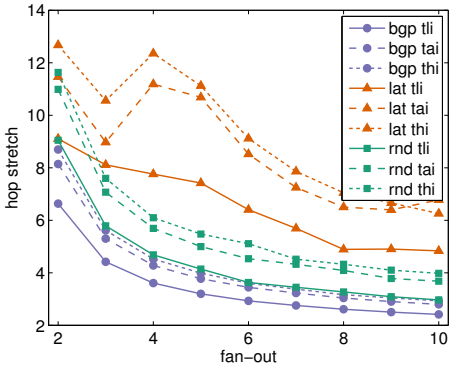
In this section we discuss the experimental evaluation results of the tree optimization strategies previously presented and afterwards look at how Lcast compares to Island Multicast, a P2P architecture that also exploits intra-domain multicast deployments.

7.1. Latency and Hop Stretch

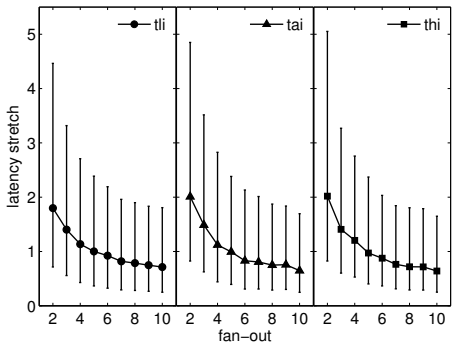
Figure 5a presents the average latency stretch for the three optimization strategies versus member fan-out. One of the first things to be noticed is the clustering of the results based on optimization strategy. On the one hand, this suggests their independence from client churn and thereby also from the size of the overlay. On the other, it indicates that the choice of the topological information to be used greatly influences latency stretch. In fact, if we split results with respect to optimization strategies, we see that *lat* outperforms *bgp* and *rnd* by a significant margin and it is generally able to ensures an average latency stretch lower than 2. Moreover, Figure 5c shows that not only the average is small but also the bounds are tight. That is, the $95th$ percentile of the latency stretch is smaller than 5 and if we focus on high fan-outs, 95% of the overlay members receive multicast content with a delay only 2 times larger than that of unicast. Since for traces like *thi* the overlay can reach up to 3000 active members, these results confirm the efficiency of the optimization algorithm and therefore Lcast's ability to deliver latency constrained multicast content. Additionally, due to the independence from churn, noticeable at least for *lat* and *rnd*, the results also indicate Lcast's adaptability to dynamic overlay conditions. This point is further supported by the observation that latency



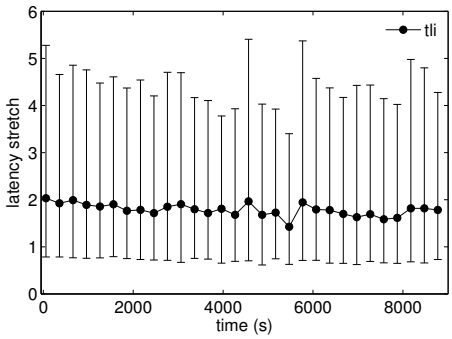
(a) Average latency stretch



(b) Average hop stretch



(c) Average latency stretch of *lat* for the three traces with 95% confidence intervals



(d) Latency stretch for *lat* with fan-out 2 and high client churn with 95% confidence intervals

Figure 5: Latency stretch and hop stretch results

stretch is rather stable with time, even when client interest is low, as depicted in Figure 5d.

As a general trend, the latency stretch values decrease with fan-out, but more importantly, increasing fan-out above 6 yields little benefit. Then, even if left unconstrained, fan-out should only seldom exceed 10 for a subset of the members. Nevertheless, such a high value might prove unacceptable in practice, therefore our decision to constrain fan-out is warranted.

A rather surprising result is the effectiveness of *rnd* relative to *bgp*. Not only are AS hops a bad estimate for latency, but using them when optimizing a distribution tree with high member fan-out yields only marginally better results than building a random tree. This, of course, questions the practicality of *bgp*. But, despite not being appropriate for minimizing latency, it will be seen later that *bgp* should be used when the aim is to minimize underlay network resource use. On the contrary, given its reasonable performance, *rnd* could be used as a

backup, no overhead optimization strategy for *lat*.

Average hop stretch results are shown in Figure 5b. The clustering with respect to optimization strategy is still noticeable however, in this case we see a clearer influence of overlay size. All results improve again with fan-out, but given the dependency of hop stretch on tree depth, fan-out here has a more important contribution. Both observations can be explained by Lcast's inability to use for replication the much better connected routers pertaining to transit domains, a typical problem for overlays. In addition, the situation is further worsened in Lcast's case by low router out degrees which prevent the optimization algorithm from taking advantage of the better connected edge routers.

Nevertheless, we see that *bgp* manages to keep hop stretch relatively low, but only once the fan-out exceeds 6. Notably, if replication factors are left unconstrained, hop stretch can drop under 3, even for large member sets and despite the imperfect BGP map used. Then, given that *bgp*'s results are a lower bound for hop stretch, it follows that Lcast will inevitably build high hop stretch paths when large number of members are joined and fan-out is kept low. The biggest disadvantage to such paths is their higher chance of being unstable as length increases, even if the inter-domain links that make them up are generally more stable than links in edge networks. So, if overlay stability is a concern, one the one hand, it could be achieved by relaxing the fan-out constraint for routers high in the distribution tree (i.e., close to the ITR), as a compensation for their lower latency stretch. On the other, it could be ensured solely through Lcast mechanisms at the cost of higher communication overhead.

Like for latency stretch, *rnd* has hop stretch close to that of *bgp* and actually performs better than *lat*. We explain the result by the fact that *rnd* generally tries to build k-ary complete (i.e. low depth) trees in a topology with a low average AS path length. In fact, since there are few inter-AS paths that could penalize overlay efficiency, *bgp*'s margins over *rnd* are not very large. In contrast, *lat* builds trees with low latency paths at the cost of higher tree depth and therefore higher overlay path lengths.

It can be noticed that for high fan-out values the latency stretch of *lat* (see Figure 5c) is lower than 1. So, the use of Lcast with *lat* as optimization strategy should result in lower average latency stretch than IP-multicast implemented with existing BGP policies. This is due to BGP's limited decision process whereby the best path is usually computed based only on AS hop distance, independent of latency. Such artefacts have been termed *latency triangle inequality violations* [54, 55] and their effect is that a subset of the BGP selected paths possess higher latency than others which, despite looking like detours due to increased number of hops, have low aggregated latency. As a result, it may happen that overlays offer at times lower inter-member latencies than the underlying unicast topology. Both [54, 55] have identified lower latency paths than the BGP selected ones for more than 20% of the pairs in their datasets. Then,

7.2. Tree Cost

Figure 6 depicts the results for tree cost. Independent of churn, results show *rnd* and *bgp* as the the worst and respectively best performer, although dif-

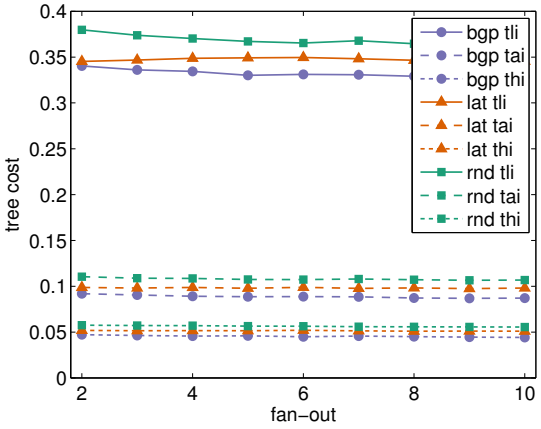


Figure 6: Tree cost with respect to member fan-out.

ferences are quite small. This corroborates our assumption that *rnd*'s previous results benefit from the low length AS paths and not from efficient opportunistic distribution trees. In contrast, we see that *lat*'s high hop stretch does not result in a high tree cost. Therefore, it follows from (1) that the long overlay paths it builds are reused by large client sets. So, in unstable network conditions, using BGP information to constrain hop stretch, i.e., building a hybrid *lat* and *bgp* optimization strategy, may be advisable. Notice however that this is not characteristic to *lat* but to all overlays that have as objective the minimization of latency alone.

The highest tree cost, registered by *rnd*, is under 0.4 for high client churn. Hence, even in the worst case, when clients are sparsely distributed within domains, Lcast requires less than half of the unicast case AS hops to deliver the content to all clients. But more importantly, for average and low client churn our solution is more than one order of magnitude more efficient than unicast.

It is also interesting to observe in Figure 6 that tree cost is independent of fan-out and barely affected by optimization strategy. Or otherwise stated, the metric is independent of the shape of the distribution tree. This is in agreement with the findings of Chalmers and Almeroth, who have previously shown that multicast efficiency, if defined similarly to our tree cost, only depends on the number of clients in the system [56]. Note though that, here, the ratio between the average tree costs for different traces is not in direct relation with their ratio of the number of clients in the overlay. This because, as it may be seen in Figure 4, the ratio changes over a simulation run. We explain the slight dependence on optimization strategy by the inefficient branching done by both *rnd* and *lat*, which does not follow underlay topology, i.e., an AS path may be crossed several times. For *bgp* this topological incongruence is minimized and considering the result above, its tree cost should be close to optimal, despite its use of an algorithm that minimizes average client AS hops to the source, not

overall bandwidth usage. Moreover, this also implies that Lcast with *bgp* should in general have a slightly smaller tree cost than other architectures focused on minimizing latency stretch.

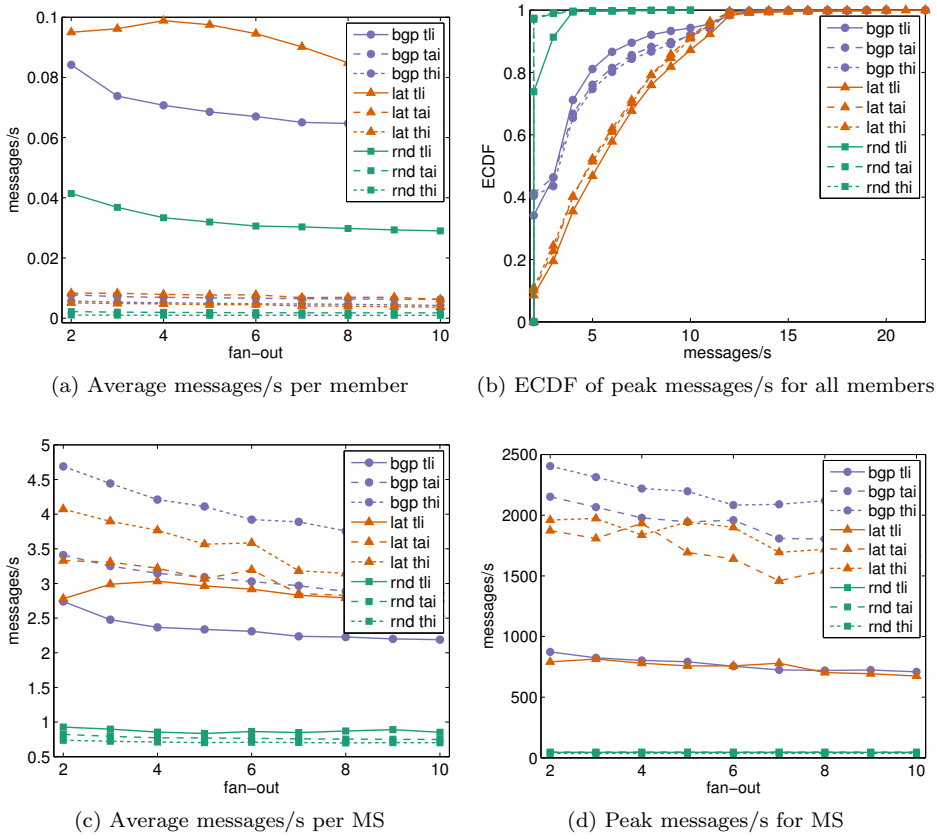
7.3. Control Traffic Overhead

We split control traffic overhead in management overhead, needed for group management due to peer churn, and active topology discovery overhead needed to perform and convey topology measurements. The only optimization strategy to employ active topology discovery is *lat*.

Figure 7a show the results for average management overhead from member perspective. The highest average rate is less that 0.11 messages/s and indicates that members seldom exchange messages with their peers or the MS. It would appear that higher churn results in higher rates but this can be attributed to the low number of overlay members. That is, for average and high client interest the overlay contains many members that seldom exchange messages so the average is kept very low. Figure 7b illustrates for each optimization strategy the Empirical Cumulative Distribution Function (ECDF) of the peak messages/s per member. For each member, the peak is computed as the maximum over all fan-outs. It may be seen that, independent of churn, members have the highest instantaneous overhead for *lat* while for *rnd* the lowest. In particular, for the former 99% of the members have peaks under 13 messages/s while for the latter the peaks are under 4 messages/s. Even in the worst recorded case, a member does not exceed 22 messages/s. We can therefore conclude that both average and instantaneous member management overhead in Lcast is negligible. An explanation for the higher number of messages exchanged when using *lat* versus *bgp* is that the overlay has a higher chance of being optimized once a new member joins or when new inter-member latencies are measured.

Looking at Figure 7c and Figure 7d, we see that the MS is also exposed to low average and instantaneous group management overhead. In fact, the highest instantaneous rate registered is 2500 messages/s and the average never goes above 5 messages/s. These message rates are easily manageable by off-the-shelf hardware. As expected, overhead is considerably higher for *bgp* and *lat* than for *rnd*, and in the case of the former two, increases with overlay size. Surprisingly, we also see that *bgp* requires slightly higher message rates than *lat*. So, although members have higher instantaneous message rates for *lat*, the MS has higher peak rates for *bgp*. This means that in general, *bgp* is more likely to update the whole topology while *lat* often performs local optimizations. Nevertheless, Figure 7c shows that both are very stable due to the small average rates.

Compared to management overhead, the active topology discovery employed by *lat* requires more involvement from the MS and members. Alas, not having implemented an optimized communication protocol between MS and members, we can not provide the exact number of messages that are exchanged. However, in the worst case, the MS would need one message exchange with a member to request that it measures one of its peers and to receive the measurement result. Although, we stress that in an optimized situation it may batch multiple



(a) Average messages/s per member

(b) ECDF of peak messages/s for all members

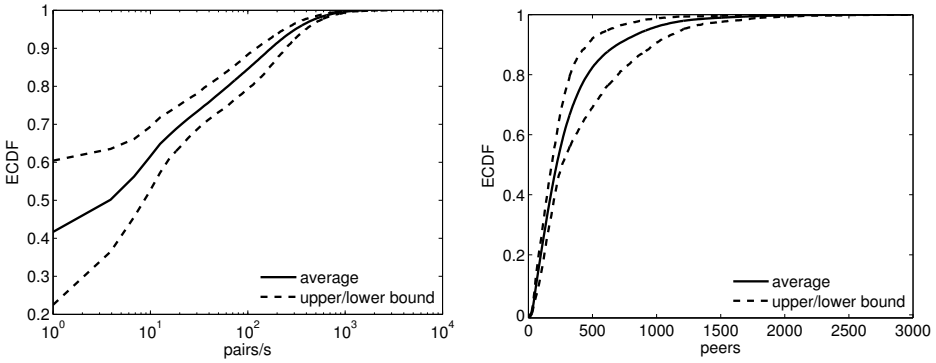
(c) Average messages/s per MS

(d) Peak messages/s for MS

Figure 7: Control traffic overhead

measurement requests in one packet. Then, considering this approximation, we can use the number of member pairs measured per second as worst case estimate of the MS's message rate. In the experiments we set k, the constant that limits the size of the cluster for our latency discovery protocol, to 5.

Figure 8a depicts the average ECDF for the number of member pair latencies measured per second, with lower and upper bounds, computed over all the *lat* simulation runs. We see that on average, during more than 41% of the time spent in a simulation, the MS does not request any measurements while in 79% to 89% of the time, less than 100 pairs are measured per second. In the worst situation, just 0.8% (or 72s) of the simulation time is spent performing more than $1k$ measurements/s. Then, typically, topology discovery overhead is negligible and in the worst case, the message rate is the same order of magnitude as peak management overhead. So, even when the two are considered together, they should still be manageable by one server. Moreover, because the MS coordinates the measurements, it is the only one affected by requests peaks. Hence, in



(a) ECDF for peer pairs measured/s (b) ECDF for peers measured/member

Figure 8: Control traffic overhead for *lat* due to active topology measurements considering all simulation runs.

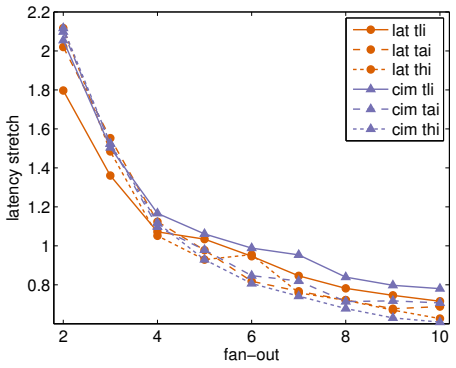
overload situations, it may queue requests for later processing with limited or no effect for the quality of the distribution tree.

Finally, Figure 8b shows that in general half of the overlay members participate in less than 220 measurements and only 4% of the members participate in more than $1k$ pairwise latency estimations over the entire length of the simulation. Thereby, the topology discovery overhead from member perspective is low, despite the large number of ASes participating in the overlay.

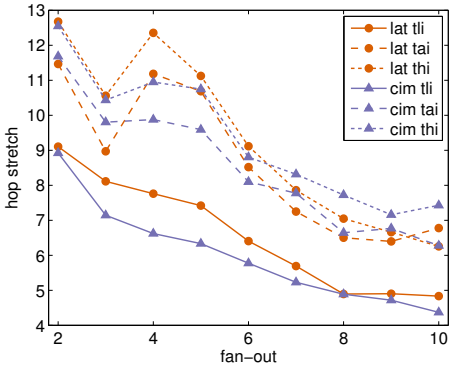
7.4. Comparison with Island Multicast

Island Multicast (IM) [7] is a P2P architecture that optimizes content delivery efficiency by exploiting intra-domain multicast deployments. Similarly to Lcast, it uses unicast to connect multicast islands however unlike Lcast, the overlay is constructed with end-hosts. IM can operate either with a centralized controller (CIM) when it optimizes the distribution tree using a variant of Shi's algorithm [42] or fully distributed (DIM) when it relies on a Delaunay triangulations (DT) overlay protocol to connect the multicast islands. Although less scalable, CIM is comparable to DIM in terms of latency stretch for large sessions, and, as shown in [57], DT is actually less efficient than overlays that take into consideration network layer latency. We therefore focus in what follows on the comparison between *lat* and CIM.

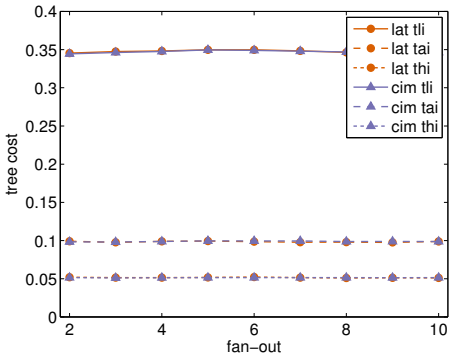
Both Lcast and CIM can control member fan-outs to limit processing and bandwidth overhead. Even though CIM can individually constrain end-host fan-outs, we perform the comparison considering fixed domain, or multicast island, out degrees and suppose that the replication load is distributed optimally over a domain's end-host population. Arguably, if fan-outs are very large, this gives a scaling advantage to CIM since all processing associated to packet replication is supported by border routers in Lcast, despite the bandwidth restrictions that must be met at border routers being the same for both solutions. Nonetheless, a



(a) Average latency stretch



(b) Average hop stretch

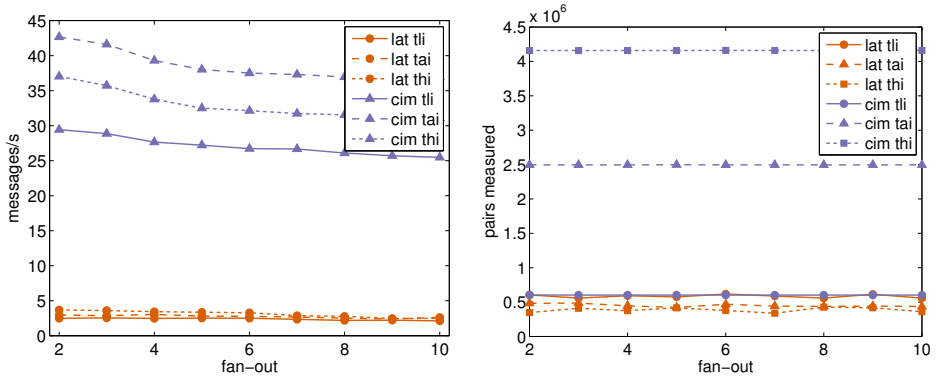


(c) Tree cost with respect to member fan-out.

Figure 9: Latency stretch, hop stretch and tree cost comparison for *lat* and CIM.

disadvantage to end-host replication is that it increases tree latencies as packets need to travel intra-domain and accumulate additional processing delays. Still, for simplicity, in our simulations we consider intra-domain propagation and end-host processing times as negligible to inter border router latencies.

Figure 9 depicts the latency stretch, hop stretch and tree cost for *lat* and CIM. Since both use the same optimization algorithm the three metrics have similar values. Particularly, in the case of latency stretch, for low and average client interest, Lcast offers slightly better performance. For *thi* however, CIM's random latency discovery algorithm benefits from end-host stability and discovers sufficient link latencies to ensure a more efficient distribution tree for high fan-outs. The fact that *lat* and CIM generate similar but not identical trees, due to their distinct approaches to latency discovery, is also supported by the average hop stretch results depicted in Figure 9b. As expected, tree cost results are identical, given their dependence on the number of clients in the system.



(a) Average messages/s for MS

(b) Number of AS pairs measured

Figure 10: Control traffic and topology discovery overhead comparison between *lat* and CIM.

An important difference between the two solution is that CIM's controller must communicate with all end-hosts participating in the overlay. As a consequence, the good performance seen in the previous comparisons comes at cost of much higher communication overhead due to peer churn. Figure 10a illustrates this point. We see that in all cases CIM requires almost one order of magnitude more messages per second than *lat* for overlay management.

In addition, one of CIM's least efficient or scalable mechanisms is the random topology discovery algorithm. This procedure is overseen by the overlay coordinator and consists of end-hosts measuring their latency to 5 random peers to discover link latencies or failed hosts. However, a clear downside to it is that if good quality distribution trees are required, measurements must be performed at intervals inversely proportional to the overlay size, that is, more often as the overlay size grows. In our simulations we set a lower bound of 20s and optimize the peer selection to ensure that inter-domain latencies are measured only once and intra-domain ones never. Figure 10b shows the total number of pairs measured and contrasts it with that of *lat*. For small overlays and high churn, the two architectures measure approximately the same number of peers however, despite our optimizations, as client churn diminishes and the overlay sizes grow, CIM requires more measurements whereas *lat* requires less. In the extreme case, for *thi*, CIM measures about an order of magnitude more peers. Considering that latency stretch is generally on par, this confirms again the ability of *lat* to efficiently manage link measurements.

Overall, the comparison shows that, if evaluated under the same constraints, *lat* and CIM optimized overlays have similar properties, save for communication overhead, which is considerably lower in *lat*'s case. This confirms *lat*'s efficient design and ability to accommodate large overlay sizes. It is also worth noting that CIM's impracticality for large overlays makes Lcast, to the best of our

knowledge, the only solution capable of ensuring on-line swapping of overlay optimization algorithms. However we would like to stress that, due to its need for infrastructure support and provided feature set, Lcast is of greater interest for network operators and thereby complementary to existing P2P streaming solutions.

8. Conclusions

Our goal with Lcast was to devise an inter-domain multicast framework that, besides possessing a low deployment cost, is also easily configurable and scalable. The former requirement was fulfilled by using just LISP enabled domain border routers to form an inter-domain overlay, without requiring any further support or changes in the Internet's core. But, equally important, by exposing the service to the clients by means of existing intra-domain multicast protocols, and by limiting the router overlay fan-out (replication factor) to low values.

Configurability was ensured by two design decisions: first, the separation of the control and data-planes and second, the centralization of the control-plane functions in the MS. Member participation in the data-plane is conditioned only by the implementation of LISP functionality. However, member presence in the control plane is not required since all optimization functions are centralized in the MS. As a result, operators may switch between tree optimization algorithms easily, even on-line, assuring fast (re)configuration of the overlay's topology to meet operational performance requirements.

The isolation through design between local-domain and inter-domain multicast allows the separation of the overlay's router members from the churn specific to client end-hosts and thus relieves the architecture's control plane from the inherent overhead. This ensures the scaling of the architecture with the number of end-hosts however, the scaling with the number of member domains is attained through proper data and control plane design.

We evaluated three possible overlay management strategies for low latency content delivery and inferred that they are all fit for optimizing large overlays. Several conclusions can be drawn from the analysis. We saw control overhead is manageable by a single server, independent of client churn and even when active topology discovery is employed. Client churn, generally, slightly influences performance but it does increase management overhead. Another very encouraging result is that Lcast's performance does not depend on large fan-outs and in fact, fan-outs larger that 6 offer limited benefits. Finally, we saw that Lcast can be used to minimize various metrics and its performance is comparable to other ALM solutions. Notably, when used with *lat*, it can deliver content with a very low, unicast like, latency in exchange for increased but still manageable control overhead.

Acknowledgements

Many thanks to Harsha Madhyastha for the help with the iPlane latency lookup service and to Damien Saucez for his valuable comments. This work has

been partially supported by the Spanish Ministry of Education under scholarship AP2009-3790, research project TEC2011-29700-C02, Catalan Government under project 2009SGR-1140 and a Cisco URP Grant.

References

[1] C. Labovitz, S. Iekel-Johnson, D. McPherson, J. Oberheide, F. Jahanian, Internet inter-domain traffic, ACM SIGCOMM Computer Communication Review 40 (4) (2010) 75–86.

[2] S. Murray, Digital TV World Revenue Forecasts (July 2012).
URL `http://www.digitaltvresearch.com/ugc/press/37.pdf`

[3] S. Deering, Host extensions for IP multicasting, RFC 1112 (INTERNET STANDARD), updated by RFC 2236 (Aug. 1989).
URL `http://www.ietf.org/rfc/rfc1112.txt`

[4] H. Holbrook, B. Cain, Source-Specific Multicast for IP, RFC 4607 (Proposed Standard) (Aug. 2006).
URL `http://www.ietf.org/rfc/rfc4607.txt`

[5] L. Zheng, Z. Zhang, R. Parekh, Survey Report on PIM-SM Implementations and Deployments, draft-zzp-pim-rfc4601-update-survey-report, work in progress (Dec. 2012).

[6] B. Zhang, W. Wang, S. Jamin, D. Massey, L. Zhang, Universal IP multicast delivery, Computer Networks 50 (6) (2006) 781–806.

[7] X. Jin, K.-L. Cheng, S.-H. Chan, Island multicast: combining IP multicast with overlay data distribution, Multimedia, IEEE Transactions on 11 (5) (2009) 1024–1036.

[8] C. Diot, B. N. Levine, B. Lyles, H. Kassem, D. Balensiefen, Deployment issues for the IP multicast service and architecture, IEEE Network 14 (1) (2000) 78–88.

[9] C. Wu, B. Li, S. Zhao, Diagnosing network-wide p2p live streaming inefficiencies, in: INFOCOM, Proceedings IEEE, 2009, pp. 2731–2735.

[10] D. Saucez, L. Iannone, O. Bonaventure, D. Farinacci, Designing a Deployable Internet: The Locator/Identifier Separation Protocol, IEEE Internet Computing 16 (2012) 14–21.

[11] B. Cain, S. Deering, I. Kouvelas, B. Fenner, A. Thyagarajan, Internet Group Management Protocol, Version 3, RFC 3376 (Proposed Standard), updated by RFC 4604 (Oct. 2002).
URL `http://www.ietf.org/rfc/rfc3376.txt`

[12] N. McKeown, Software-defined Networking, Keynote Talk at IEEE INFOCOM 2009.

[13] Sopcast P2P Internet TV.
URL http://www.sopcast.com

[14] H. Eriksson, MBONE: the multicast backbone, Communications of the ACM 37 (8) (1994) 54–60.

[15] G. Bumgardner, Automatic Multicast Tunneling, draft-ietf-mboned-auto-multicast-14, work in progress (Jun. 2012).

[16] S. Banerjee, B. Bhattacharjee, C. Kommareddy, Scalable application layer multicast, in: Proceedings ACM SIGCOMM, 2002.

[17] Y. hua Chu, S. Rao, S. Seshan, H. Zhang, A case for end system multicast, Selected Areas in Communications, IEEE Journal on 20 (8) (2002) 1456 – 1471.

[18] S. Banerjee, C. Kommareddy, K. Kar, B. Bhattacharjee, S. Khuller, Construction of an efficient overlay multicast infrastructure for real-time applications, in: Proceedings of IEEE INFOCOM, 2003, pp. 1521–1531.

[19] D. Tran, K. Hua, T. Do, A peer-to-peer architecture for media streaming, Selected Areas in Communications, IEEE Journal on 22 (1) (2004) 121 – 133.

[20] M. Castro, P. Druschel, A.-M. Kermarrec, A. Rowstron, Scribe: a large-scale and decentralized application-level multicast infrastructure, Selected Areas in Communications, IEEE Journal on 20 (8) (2002) 1489 – 1499.

[21] PPLive P2P Internet TV.
URL http://www.pplive.com/

[22] X. Zhang, J. Liu, B. Li, Y.-S. Yum, Coolstreaming/donet: a data-driven overlay network for peer-to-peer live media streaming, in: INFOCOM, Proceedings IEEE, Vol. 3, 2005, pp. 2102 – 2111.

[23] UUSee P2P Internet TV.
URL http://www.uusee.com/

[24] X. Hei, C. Lian, J. Lian, Y. Liu, K. W. Ross, A Measurement Study of a Large-Scale P2P IPTV System, TOM 9 (8).

[25] L. Vu, I. Gupta, J. Liang, K. Nahrstedt, Measurement and modeling of a large-scale overlay for multimedia streaming, QSHINE.

[26] T. Silverston, L. Jakab, A. Cabellos-Aparicio, O. Fourmaux, K. Salamatian, K. Cho, Large-scale measurement experiments of P2P-TV systems insights on fairness and locality, Signal Processing: Image Communication 26 (7) (2011) 327 – 338.

[27] E. Veloso, V. Almeida, W. Meira, A. Bestavros, A hierarchical characterization of a live streaming media workload, Networking, IEEE/ACM Transactions on 14 (1) (2006) 133–146.

[28] L. Jakab, A. Cabellos-Aparicio, T. Silverston, M. Sole, F. Coras, J. Domingo-Pascual, Corecast: How core/edge separation can help improving inter-domain live streaming, Computer Networks 54 (18) (2010) 3388 – 3401.

[29] D. Farinacci, D. Meyer, J. Zwiebel, S. Venaas, The Locator/ID Separation Protocol (LISP) for Multicast Environments, RFC 6831 (Experimental) (Jan. 2013).
URL `http://www.ietf.org/rfc/rfc6831.txt`

[30] D. Meyer, L. Zhang, K. Fall, Report from the IAB Workshop on Routing and Addressing, RFC 4984 (Informational) (Sep. 2007).
URL `http://www.ietf.org/rfc/rfc4984.txt`

[31] LISP Testbed.
URL `http://www.lisp4.net/`

[32] D. Farinacci, V. Fuller, D. Meyer, D. Lewis, The Locator/ID Separation Protocol (LISP), RFC 6830 (Experimental) (Jan. 2013).
URL `http://www.ietf.org/rfc/rfc6830.txt`

[33] R. Hinden, New Scheme for Internet Routing and Addressing (ENCAPS) for IPNG, RFC 1955 (Informational) (Jun. 1996).
URL `http://www.ietf.org/rfc/rfc1955.txt`

[34] V. Fuller, D. Farinacci, D. Meyer, D. Lewis, Locator/ID Separation Protocol Alternative Logical Topology (LISP+ALT), RFC 6836 (Experimental) (Jan. 2013).
URL `http://www.ietf.org/rfc/rfc6836.txt`

[35] V. Fuller, D. Lewis, V. Ermagan, A. Jain, LISP Delegated Database Tree, draft-fuller-lisp-ddt-04.txt, work in progress (Sep. 2012).

[36] F. Coras, A. Cabellos-Aparicio, J. Domingo-Pascual, F. Maino, D. Farinacci, Locator/ID Separation Protocol (LISP), draft-coras-lisp-re-02, work in progress (Feb. 2013).

[37] N. Feamster, H. Balakrishnan, J. Rexford, A. Shaikh, J. Van Der Merwe, The case for separating routing from routers, in: Proceedings of the ACM SIGCOMM workshop on Future directions in network architecture, ACM, 2004, pp. 5–12.

[38] S. Yeganeh, A. Tootoonchian, Y. Ganjali, On scalability of software-defined networking, Communications Magazine, IEEE 51 (2) (February) 136–141.

[39] D. Farinacci, M. Napierala, LISP Control-Plane Multicast Signaling, draft-farinacci-lisp-mr-signaling-01, work in progress (Jan. 2013).

[40] H. Holbrook, B. Cain, B. Haberman, Using Internet Group Management Protocol Version 3 (IGMPv3) and Multicast Listener Discovery Protocol Version 2 (MLDv2) for Source-Specific Multicast, RFC 4604 (Proposed Standard) (Aug. 2006).
URL http://www.ietf.org/rfc/rfc4604.txt

[41] B. Fenner, M. Handley, H. Holbrook, I. Kouvelas, Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised), RFC 4601 (Proposed Standard), updated by RFCs 5059, 5796, 6226 (Aug. 2006).
URL http://www.ietf.org/rfc/rfc4601.txt

[42] S. Shi, J. Turner, M. Waldvogel, Dimensioning server access bandwidth and multicast routing in overlay networks, in: NOSSDAV, Proceedings ACM, 2001, pp. 83–91.

[43] H. V. Madhyastha, T. Isdal, M. Piatek, C. Dixon, T. Anderson, A. Krishnamurthy, A. Venkataramani, iPlane: An information plane for distributed services, in: USENIX OSDI, 2006.

[44] Routeviews project.
URL http://www.routeviews.org

[45] Y. Hyun, B. Huffaker, D. Andersen, E. Aben, M. Luckie, kc claffy, C. Shannon, The IPv4 Routed /24 AS Links Dataset - 2011-04.
URL http://www.caida.org/data/active/ipv4_routed_topology_aslinks_dataset.xml

[46] RIPE, Routing Information Service (RIS).
URL https://labs.ripe.net/datarepository/data-sets/routing-information-service-ris-raw-data-set

[47] M. Faloutsos, P. Faloutsos, C. Faloutsos, On power-law relationships of the internet topology, in: SIGCOMM, 1999.

[48] X. Dimitropoulos, D. Krioukov, G. Riley, Revisiting Internet AS-Level Topology Discovery, in: C. Dovrolis (Ed.), Passive and Active Network Measurement, Vol. 3431 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2005, pp. 177–188.

[49] G. Huston, AS6447 BGP routing table analysis report.
URL http://bgp.potaroo.net/as6447

[50] H. V. Madhyastha, E. Katz-Bassett, T. Anderson, A. Krishnamurthy, A. Venkataramani, iPlane Nano: path prediction for peer-to-peer applications, in: NSDI, Proceedings USENIX, 2009, pp. 137–152.

[51] L. Jakab, A. Cabellos-Aparicio, F. Coras, D. Saucez, O. Bonaventure, LISP-TREE: A DNS Hierarchy to Support the LISP Mapping System, Selected Areas in Communications, IEEE Journal on 28 (8) (2010) 1332 –1343.

[52] F. Coras, T. Silverston, J. Domingo-Pascual, A. Cabellos-Aparicio, A Measurement Study of SOPCast, Tech. Rep. UPC-DAC-RR-CBA-2012-2. URL http://personals.ac.upc.edu/fcoras/publications/2012-fcoras-sopcast-study.pdf

[53] K. Sripanidkulchai, B. Maggs, H. Zhang, An Analysis of Live Streaming Workloads on the Internet, in: IMC, 2004.

[54] S. Savage, A. Collins, E. Hoffman, J. Snell, T. Anderson, The end-to-end effects of internet path selection, ACM SIGCOMM Computer Communication Review 29 (4) (1999) 289–299.

[55] C. Lumezanu, R. Baden, N. Spring, B. Bhattacharjee, Triangle inequality and routing policy violations in the internet, in: PAM, Springer, 2009, pp. 45–54.

[56] R. Chalmers, K. Almeroth, On the topology of multicast trees, Networking, IEEE/ACM Transactions on 11 (1) (2003) 153–165.

[57] J. Liebeherr, M. Nahas, W. Si, Application-layer multicasting with delaunay triangulation overlays, Selected Areas in Communications, IEEE Journal on 20 (8) (2002) 1472–1488.