

ISP-friend or foe? Making P2P live streaming ISP-aware

Fabio Picconi and Laurent Massoulie
Thomson, France
{fabio.picconi,laurent.massoulie}@thomson.net

Thomson Technical Report
Number: CR-PRL-2008-11-0001
Date: November 27, 2008

Abstract: Current peer-to-peer systems are network-agnostic, often generating large volumes of unnecessary inter-ISP traffic. Although recent work has shown the benefits of ISP-awareness on bulk transfer applications, no studies have focused on optimizing P2P live streaming systems. These are harder to design, as data must be diffused to all receivers within short delays.

In this paper we propose a novel scheme for ISP-friendly mesh-based live streaming. Each peer maintains two distinct sets of overlay neighbors, used respectively for local and global stream propagation. A dynamic unchoke mechanism minimizes inter-ISP traffic in normal operation, enabling it promptly when local diffusion is impaired, e.g., when fast local sources become suddenly unavailable. Our scheme is independent of the chunk scheduling algorithm, and thus can be applied to a wide range of existing systems.

We have integrated our ISP-friendly scheme to our P2P live streaming prototype, and evaluated its performance through emulation and Planetlab experiments. Our results show that our scheme adapts quickly to churn and network partitions, and achieves up to a ten-fold reduction in transit traffic.



1. Introduction

Peer-to-peer applications have become the dominant source of Internet traffic, surpassing by far web and e-mail. In 2004, P2P file sharing applications such as BitTorrent, Kazaa and eDonkey were already generating more than 50% of the Internet's backbone traffic [1]. Today, file sharing is still extremely popular, and P2P video streaming is on the rise. In 2007, PPLive [2], a P2P-video company, reported an average of 1.5 million active users [3]. Many other systems, such as TVants [4] and SopCast [5], are also widely used on the Internet.

While P2P has been widely welcome by end-users, it has also become a headache for Internet Service Providers (ISPs). Current P2P algorithms are network-agnostic, i.e., they construct an overlay without any knowledge of the underlying network topology. As a result, peers often exchange data with users located on different ISPs, instead of favoring those connected to the local ISP. This behavior produces large amounts of unnecessary inter-domain traffic [8], generating a high financial cost for ISPs.

A solution to this problem consists in biasing overlay neighbor selection using the information obtained from an ISP-aware service [9, 10, 11]. This leads to clustered overlays which closely follow the underlying physical topology. As a consequence, inter-domain traffic is significantly reduced, lowering the financial burden for ISPs and freeing up wide-area bandwidth for end-users. However, this approach has only been evaluated with bulk-transfer applications, e.g., BitTorrent [6], whose traffic is elastic and thus can be manipulated rather easily. Conversely, streaming applications must ensure a minimum sustained rate, and, more importantly, must deliver packets to all receivers within a short delay. Thus, any streaming traffic optimizations must also ensure that the QoS is not impacted.

In this paper, we propose a novel mechanism that reduces inter-ISP traffic in P2P live streaming applications. Unlike previous schemes [9, 10], our system creates a large number of links between peers located in different ISPs. This results in a highly clustered primary overlay, augmented by a large number of secondary inter-cluster links. Moreover, secondary links are dynamically unchoked to provide both high network efficiency and good reactivity to sudden network changes.

We show analytically that this scheme allows near-optimal diffusion provided inter-cluster rates are adjusted appropriately. We propose a heuristic that approximates these rates by having each peer adjust its unchoke level according to the contents of its local download buffer. We evaluate the performance of our prototype by deploying up to 320 clients in an emulated environment [14] and on Planetlab [15]. Our results show that our overlay quickly converges to a highly clustered topology, reducing inter-

ISP traffic by up to a factor of 10 for a 4-ISP topology. Moreover, our dynamic unchoke algorithm provides quick adaptation to network partitions, reducing the probability of missing playback deadlines.

This paper makes the following contributions: 1) we present a novel mechanism for ISP-friendly mesh-based live streaming, which is scheduling-independent and thus can be applied to a wide range of existing systems, 2) we show analytically that our scheme can achieve near-optimal diffusions, 3) we validate our design by performing an experimental evaluation of our prototype.

This paper is organized as follows. Section 2 describes related work. Section 3 presents our design and Section 4 some implementation details. Section 5 contains the results of our experimental evaluation, and Section 6 concludes the paper.

2. Related work

Recent studies have shown that network-agnostic P2P systems can generate large amounts of inter-domain traffic. Karagiannis et al. [8] analyzed BitTorrent traffic at the access-link of a university network of 20,000 users. They found that 50 to 90% of data chunks are downloaded from external sources despite being locally available. Hei et al. [25] performed a measurement study of PPLive [2], a widely-deployed P2P live streaming system. They deployed peers in North America, and observed that 70-90% of the download traffic originated from peers located in Asia.

Several research efforts have been directed towards reducing inter-ISP traffic. Shen et al. [7] propose to cache P2P content in ISP-controlled devices. While effective, P2P caches need to be designed for each application, and are not without legal risks for the ISP due to unauthorized exchange of copyrighted material.

A different approach consists in biasing a peer's neighbor selection towards topologically close nodes. Aggarwal et al. [9] propose an Oracle server that returns a list of peers located close to the requestor. Their study is mainly based on simulation, and focuses on file sharing applications. Choffnes et al. [11] suggest leveraging existing CDNs, which are already network-aware, instead of deploying ad-hoc servers. They evaluate their solution by providing an extended version of Azureus, a popular file-sharing client, and collecting measurements of its real behavior.

A similar but more sophisticated scheme has been recently proposed by Xie et al. [10]. They suggest using an ISP-controlled tracker that provides a richer set of network-related information, and performs traffic-engineering optimizations. Although their evaluation focuses on file-sharing, the authors include one live streaming experiment which shows that backbone traffic is reduced by a factor of 2.5.

Lua et al. [28] propose a network-aware live streaming system where the overlay is constructed according to the network latency between peers, i.e., the Round-Trip-Time. Since RTTs do not necessarily capture the relationships between ISPs, their design may generate excessive inter-ISP traffic.

3. Design

In this Section we first introduce the goals and rationale of our design, and then describe the details of our overlay construction and dynamic unchoke algorithms.

3.1. Goals and design rationale

The goal of an ISP-friendly live streaming system is to minimize inter-ISP traffic without impacting the video quality. This is not easy to achieve. First, the minimizing inter-ISP traffic may generate excessively long diffusion delays. Consider an example topology with N peers distributed among n ISPs, where ISPs i and $i+1$ are connected by peering links (i.e., ISPs form a chain of peering links), and any two ISPs are reachable through more expensive backbone links. In this case, the minimum-cost diffusion follows the peering links, and thus results in a maximum delay of order n . In general, a scheme that generates some backbone traffic can significantly lower diffusion delay at the cost of a suboptimal network cost. Second, centrally-computed traffic optimizations such as that performed by P4P [10] may be impractical in live streaming systems, where peers must react quickly to avoid deadline misses in the presence of churn and network partitions. Thus, quick local decision may be more effective than centralized control schemes.

Our design is based on the following principles: (1) we favor simplicity and low diffusion delays over traffic optimality, (2) we employ decentralized mechanisms based on local peer decisions, (3) we target mesh-based systems, but we remain agnostic to the scheduling algorithm, (4) we limit centralized services to a tracker node that keeps the list of active peers, and an ISP-managed server that reports the network cost between any two IP addresses [9, 10]. However, we do not require these services to be centralized. The tracker could be implemented in a decentralized way, e.g., using a DHT [12], and the network cost could be estimated in a distributed fashion by reverse-engineering the network topology [20, 21].

Our solution is based on a combination of two novel mechanisms: a randomized two-level overlay, and a dynamic unchoke mechanism for costly traffic. Our overlay creates clusters of topologically-close peers, typically within the same ISP, while still preserving a large number of links between peers located in distant points of the network. The unchoke mechanism dynamically adapts the bandwidth

of inter-cluster links to ensure that chunks are diffused globally with minimal inter-ISP traffic. In Section 3.4 we show that the combination of these two mechanisms can lead to near-optimal diffusions, provided the unchoke level is adjusted appropriately. We now describe both mechanisms in more detail.

3.2. Overlay construction

We assume that the network cost between any two nodes is known. We construct a directed overlay containing two types of edges: primary and secondary. The former are created preferentially between nearby peers (i.e., network cost between them is low), while the latter connect peers at random, irrespective of network cost. Both the maximum primary and secondary out-degrees of a peer are set to $d_{o,max} := \lfloor C/c_e \rfloor$, where C is the peer’s upload capacity, and c_e a system-wide nominal edge capacity¹. The maximum primary and secondary in-degree is set to $d_{i,max} := \lceil r/c_e \rceil$, where r is the stream rate².

The objectives of the overlay formation are then as follows. In the primary overlay, one should (i) let each peer have in-degree of $d_{i,max}$, and out-degree no larger than $d_{o,max}$, a peer-specific bound; (ii) minimize the sum of network costs of all primary edges; (iii) let the overlay be randomly, uniformly distributed under the previous constraints. The design objective for the secondary overlay is exactly the same, except that we drop requirement (ii). As further discussed in Section 3.4, this combination of randomization and localization enables to leverage low cost connections, while ensuring required connectivity properties.

To create such a randomized overlay in a distributed fashion, we can use the following version of the Metropolis algorithm [22]. Each peer periodically offers to create an edge towards a randomly selected peer. The latter accepts, eventually dropping an existing incoming edge, with some probability that depends on whether the cost is lowered by this edge replacement. In addition, each peer periodically drops an existing edge with a small probability. One can establish, along the lines of [29], that for suitable probabilities of transitions, this mechanism converges to the desired randomized overlay. We omit the argument for lack of space.

We implement a variant of the algorithm mentioned above. The differences are motivated by practical considerations, mainly to avoid excessively long convergence times, and to allow joining nodes to create edges quickly in order to minimize video start-up time. We now outline the most important differences.

Each peer obtains a list of other peers from an ISP-aware tracker [9, 10]. For large overlays, the list contains a sub-

¹Using a nominal edge capacity helps create a bandwidth-balanced overlay. We do not actually rate-limit edge traffic to the nominal value.

²In variable-bitrate streams, the average bitrate can be used.

set of random peers, plus a subset of low-cost ones (i.e., located near the requester). Both lists are randomized, so successive tracker queries may return different peer sets. The peer then periodically contacts a peer from the list to evaluate whether a new primary or secondary edge can be established. Peers direct half of their connection attempts to peers from the random peer list, and half to peers picked from the low-cost one, thus ensuring that candidates for primary and secondary edges are contacted. After a handshake phase, both peers agree on whether to create an edge between or not, eventually dropping an existing edge. Joining nodes are handled differently. Peers always accept edge creation requests from joining nodes, thus ensuring their quick insertion into the overlay. This typically results in expensive edges being created, which are then progressively replaced by lower-cost ones.

3.3. Dynamic secondary unchoke

The dynamic unchoke mechanism has two goals. First, it attempts to keep secondary edge traffic as low as possible, in order to approximate an optimal cost diffusion. Second, it quickly unchokes secondary edges when a peer approaches a state of chunk starvation, in order to minimize the probability of a deadline miss.

Each peer adapts its secondary receive rate according to the presence or absence of events that suggest the peer is heading towards chunk starvation. We refer to such events as *early starvation signals*, or ESS. Whenever an ESS is generated, the peer increases its secondary receive rate. Conversely, if no ESS is observed during a given time interval T_{noESS} , the secondary rate is decreased. This produces quick secondary unchokes, but slower conservative chokes. We adjust the secondary rate using multiplicative-increase/multiplicative-decrease with factors α and β respectively.

In our design, a peer generates an ESS whenever a chunk has not been received half-way to the deadline. For instance, when using a 20-second download buffer, a chunk that has not been received 10-seconds before the deadline triggers an ESS. Thus, we call this event a *mid-buffer miss*. The rationale is that the peer still has another 10-seconds to unchoke secondary sources and download chunks from them.

The advantage of using a mid-buffer miss as ESS is its simplicity. The disadvantage is that the download buffer length, which determines the video lag with respect to the source, must be sufficiently large to deliver all chunks before the mid-buffer position. Thus, our scheme increases the video lag compared to an ISP-agnostic system. However, this lag increase could be reduced by employing more sophisticated ESS mechanisms, such as detecting an increase in the delay distribution of all received chunks.

3.4 Cost efficiency

We now argue that the randomized, two-level overlay construction combined with the associated choking mechanism is rich enough to allow cost-efficient streaming.

Assume a two-level cost structure, with a low cost per byte for intra-ISP transfer, \underline{c} , and a higher cost per byte \bar{c} for inter-ISP transfer. We denote peers by i, j, \dots , and ISP's by I, J, \dots . The full collection of ISP's is denoted by \mathcal{J} .

Let $d_{i,max}$ denote the streaming rate, expressed in units of "nominal edge capacity". For each peer i , denote by λ_{0i} the rate at which the source injects data directly into peer i . We assume that these injection rates sum up to b , or equivalently, the source injects data only once in the system. If the source has extra capacity to send redundant data, for such additional transmissions the source is considered as an ordinary peer.

For any ISP J , we denote by $n(J)$ the number of receivers within J , and by λ_{0J} the aggregate source injection rate into J , that is:

$$\lambda_{0J} := \sum_{i \in J} \lambda_{0i}.$$

In this setup, if we take the source injection rates as given, then the minimal cost required to stream to all receivers within an ISP J is no less than:

$$c(J) = (d_{i,max} - \lambda_{0J})\bar{c} + (n(J) - 1)d_{i,max}\underline{c}. \quad (1)$$

Indeed, all the content must reach ISP J , hence data must reach J from other ISP's (hence at a cost \bar{c}) at a rate no less than $d_{i,max} - \lambda_{0J}$. In addition, all $n(J)$ receivers within J must receive data at rate $d_{i,max}$; thus an additional rate of $(n(J) - 1)d_{i,max}$ must be used to upload to these receivers, but potentially at the lower cost \underline{c} . Thus, the total cost is no larger than

$$c^* := \sum_J c(J) = d_{i,max}\bar{c}[|\mathcal{J}|-1] + d_{i,max}\underline{c}[n-|\mathcal{J}|]. \quad (2)$$

In this setup, we have the following result.

Proposition 1. *Assume that each peer has an uplink bandwidth of exactly $d_{i,max}$. Consider a symmetric scenario where for each ISP J has $n/|\mathcal{J}|$ receivers, and the source injects at the same rate $\lambda_{0J} = d_{i,max}/|\mathcal{J}|$ into each ISP J . Then for large system size n and randomized overlays as specified in Section 3.2, the following holds.*

With high probability, for some uniform choking of secondary peer sets, streaming remains feasible at cost no larger than $c^ + o(d_{i,max}|\mathcal{J}|\bar{c})$, where c^* is as defined in (2).*

If we relax the symmetry assumption, and let $n(J) = p(J)n$ for some fixed probability n , then the following

holds. With high probability, for some uniform choking of secondary peer sets, streaming remains feasible at cost no larger than $c^* + O(d_{i,max}|\mathcal{J}|\bar{c})$, where c^* is as defined in (2).

In the previous statement, the constant in the $O(\cdot)$ term involves the probabilities $p(J)$, and increases with asymmetry in these proportions. The key message is that the extra cost incurred due to cross-ISP traffic increases linearly with the number of ISP's but does not depend on the number of peers themselves.

A complete proof is provided in the Appendix.

4. Prototype

We have integrated the mechanisms described in the previous Section to our live streaming prototype, which we used in a previous experimental study [18]. We modified the code that handles overlay construction and rewiring, and added a few lines to the chunk scheduling code to limit the number of chunks sent through secondary links. We maintain the DP/LU scheduling algorithm [17], and an ISP-agnostic source that injects chunks preferentially to fast peers [18].

We implement the dynamic secondary unchoke scheme as follows. Each peer divides time into 10-second rounds, and keeps a variable s_{max} , representing the maximum number of chunks that it can download from secondary sources within a round. The value of s_{max} determines the secondary unchoke level, and is adjusted according to the MIMD algorithm described in the previous Section. Once a peer has downloaded the s_{max} chunks, it rejects any further upload tokens from secondary sources until the round is over. Note that this mechanism may concentrate a peer's secondary traffic in the beginning of the round. However, since round boundaries are not synchronized among peers, this has a negligible effect on chunk propagation.

Our scheme requires peers to know their upload capacity in order to determine their maximum out-degree. We currently set the peer's emulated capacity in its configuration file. This corresponds to the common practice of letting peer-to-peer users specify an upload limit in their client's configuration. An alternative could consist in using bandwidth estimation techniques [23] to dynamically assess the available bandwidth.

5. Evaluation

We evaluate our prototype using both network emulation [14] and Planetlab [15] experiments. Emulation allows us to perform reproducible experiments in a controlled environment, while Planetlab deployments represent more realistic and challenging conditions.

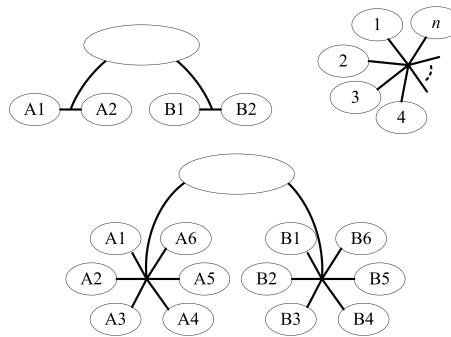


Figure 1. Test topologies: 2+2 (top left), n -clique (top right), and 6+6 (bottom)

		fraction of peers within the group
A1	B1	46.5%
A2	B2	21.4%
A3	B3	18.6%
A4	B4	6.0%
A5	B5	5.0%
A6	B6	2.5%

Table 1. ISP popularity (6+6 topology).

When using emulation, we run a 320-peer overlay on our local cluster, and emulate wide-area latencies and access bandwidths using Modelnet [14]. Our cluster consists of 10 machines with four 2.0 GHz Opteron cores each. We observe around 25% of idle CPU time, so the impact of running several clients per CPU core is negligible.

We set the clients' uplink capacities according to three different distributions. In the first, all clients have a homogeneous capacity of 1 Mbps. In the second, clients are either fast (1.5 Mbps) or slow (500 kbps), and the average capacity is 1 Mbps. In the third distribution, clients are divided into four classes: 128, 384, 1000, and 4000 kbps. This distribution is based on a measurement study and has been used in recent live streaming evaluations [16, 18]. Its mean capacity is around 1 Mbps. We set all downlink capacities to 10 Mbps.

In our Planetlab experiments we limit the upload bandwidth at the application layer using a token bucket. Each token corresponds to a chunk ready to be uploaded, and the bucket is filled at a rate proportional to the node's upload capacity. Since most Planetlab machines are usually overloaded, we limit the overlay size to 160 peers running on machines that report at least 5% idle CPU time. We also limit the upload capacities to 1 Mbps to avoid bandwidth bottlenecks.

We use three different synthetic network topologies,

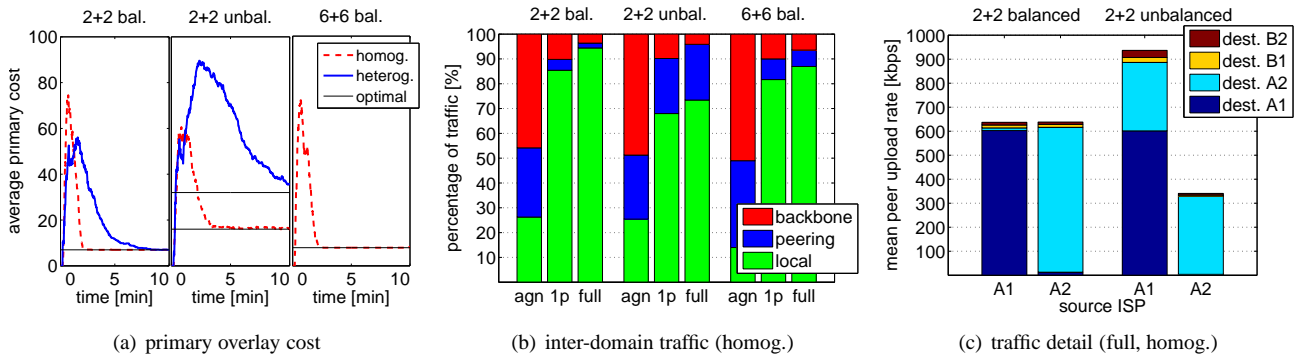


Figure 2. (a) primary overlay quickly converges to near-optimal configuration. (b) our solution generates less inter-domain traffic than the agnostic and 1-peer schemes. (c) slow peers download from least-cost ISPs in the unbalanced configuration.

shown in Figure 1. In the 2+2 topology, we have four ISPs divided into two groups, A and B. The ISPs within each group have established a peering agreement to avoid transit traffic. However, any traffic between groups must be routed through a common backbone (a higher-tier ISP). When using this topology we distribute clients uniformly across all four ISPs. Given its simplicity, this topology is useful for a baseline evaluation of our prototype. The 6+6 topology is a coarse approximation of two geographically distant countries. Each country has 6 ISPs of different sizes, all connected by peering links. Traffic between the two countries goes through a higher-tier ISP. Within each group, we distribute clients according to the ISP sizes shown in Table 1, which correspond to the market shares of the 6 main ISPs in France as of June 2007 [24]. Thus, this topology allows us to evaluate the impact of ISPs containing a small number of peers. Finally, the n -clique topology consists of n ISPs of equal size connected by pair-wise links. An evaluation with more complex topologies is left for future work.

Our overlay construction algorithm requires a network cost value c for each pair of nodes. We assume a simple cost model, where intra-ISP traffic is cheaper than peering, which in turn is cheaper than backbone. We further assume that the cost does not depend on the source and destination ISP, but only on the link type. We use $c_l = 1$ for any pair of nodes within the same ISP, $c_p = 10$ for nodes located across a peering link, and $c_b = 100$ for nodes communicating through the backbone. Note that any other values that satisfy $c_l < c_p < c_b$ produce the same overlay configurations. Peers obtain the cost value for a given destination node from the tracker.

Unless otherwise noted, we use a stream rate of 640 kbps, a source upload capacity of 1 Mbps, a chunk size of 10 KB, a download buffer of 20 seconds, a 320-peer overlay, $\alpha = 1.25$, $\beta = 0.75$, and $T_{noESS} = 10$ seconds. We set the nominal edge capacity to 100 kbps, which yields a

topol.	intra-group	intra-ISP	upload capacities [kbps]	
			ISP A1,B1	ISP A2,B2
2+2	balanced	homog.	1000	1000
	balanced	heterog.	128-4000	128-4000
2+2	unbalanced	homog.	1500	500
	unbalanced	heterog.	1000-4000	128-384
all ISPs				
6+6	balanced	homog.	1000	
n -clique	balanced	homog.	1000	

Table 2. Test scenarios.

primary in-degree of 7 for a 640-kbps stream. All experiments last up to 15 minutes, with all peers simultaneously joining the overlay at $t = 0$.

5.1. Overlay construction

We start by evaluating our overlay construction algorithm. We use the first five scenarios of Table 2. In *bandwidth-balanced* scenarios, the average client upload capacity is 1 Mbps for all ISPs. Since this average capacity is higher than the stream rate (640 kbps), chunks can be propagated within each ISP without any help from external uploaders. Conversely, in *bandwidth-unbalanced* scenarios, the average capacity of some ISPs is lower than the stream rate. Therefore, clients located in these ISPs must download part of the stream from other ISPs to sustain continuous playback. These scenarios are further divided into *homogeneous* and *heterogeneous*. In the former, all peers within an ISP have the same upload capacity, whereas the latter contain ISPs with both fast and slow peers.

Every 10 seconds we measure the cost of each peer's primary peerset, defined as the sum of the cost of its primary incoming edges. We then calculate the average value

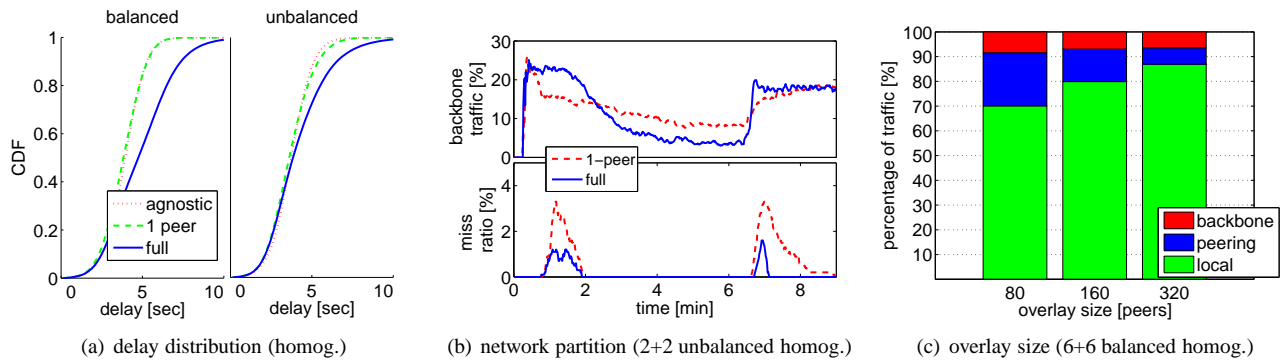


Figure 3. (a) the unchoke mechanism increases the diffusion delay to the mid-buffer position. (b) a large secondary peerset improves the system’s response to a network partition. (c) small ISPs produce less inter-ISP traffic for larger overlays.

over all peers, which represents the average cost of the overlay. We also compare this value to the optimal cost, which we obtain by solving a binary integer program with MATLAB [19].

Figure 2(a) shows the results for the different scenarios. In all cases, the overlay converges to a near-optimal configuration within a few minutes. The initial high cost is due to peers quickly populating their peersets with randomly chosen nodes, and subsequently replacing them with lower cost ones. Interestingly, the scenarios with intra-ISP homogeneous bandwidths converge more quickly than those with heterogeneous ones. Similarly, balanced scenarios show faster convergence than unbalanced ones. The reason is that convergence is slower in the presence of slow peers, which quickly allocate all their uplink capacity, and reject any further queries to become primary uploaders. The effect is more pronounced in unbalanced scenarios, where some ISPs contain *only* slow peers. Clearly, the convergence speed could be improved by using an optimized peer discovery algorithm, e.g., one that targets peers which have low cost and are known to have some available upload capacity.

5.2. Comparison to agnostic and unchoked 1-peer secondary

We now compare our solution to a simpler scheme that uses a secondary of only a single peer and no rate limiting, and to an ISP-agnostic overlay. We refer to them as “full”, “1-peer”, and “agnostic” respectively. We test each scheme in the three homogenous scenarios described before. In each case, we let the overlay stabilize for five minutes, and then measure the average backbone, peering, and local traffic for two minutes. The results are shown in Figure 2(b). We can see that the agnostic scheme produces around 50% of backbone traffic in all three scenarios. This is expected,

as both topologies have an equal number of clients in each ISP group (A and B). The higher peering traffic observed in the 6+6 topology is due to the presence of small ISPs. Since agnostic clients choose peers at random, clients in small ISPs will most likely choose peers located in other ISPs, thus generating more peering traffic.

Figure 2(b) also shows that both the 1-peer and the full scheme achieve significant reductions of backbone and peering traffic. However, the 1-peer scheme is less efficient in all three scenarios. This shows that using several dynamically choked connections to non-local peers is more efficient than a single unchoked connection. Notice that the full scheme is particularly efficient in the 2+2 balanced scenario, generating only 5% of non-local traffic. The unbalanced scenario generates more peering traffic, as clients in slow ISPs must download part of stream from fast ISPs. This is shown in more detail in Figure 2(c), where we see that peers in ISP A1 upload an average of 300 kbps to ISP A2 in the unbalanced case. These peers are exceeding the 100-kbps nominal edge capacity, since each peers in ISP A1 creates, in average, two outgoing primary edges to peers in ISP A2. This is due to the absence of rate limiting in primary edges. Enforcing a bandwidth limit on primary edges would probably reduce inter-ISP traffic in this scenario, but might also impact the diffusion delay. We leave an evaluation of primary rate limiting for future work.

We now turn our attention to the chunk diffusion delay of the three schemes. We take the two 2+2 homogeneous scenarios, and we measure the diffusion delay of all the chunks delivered within two minutes. Figure 3(a) shows the resulting Cumulative Distribution Function (CDF). The agnostic and 1-peer scheme have almost identical delay distributions, suggesting that the clustered overlay structure of the 1-peer scheme does not have an impact on delay. The full scheme, in turn, shows an increase in delay, with a significant number of chunks experiencing delays close to 10 seconds. No-

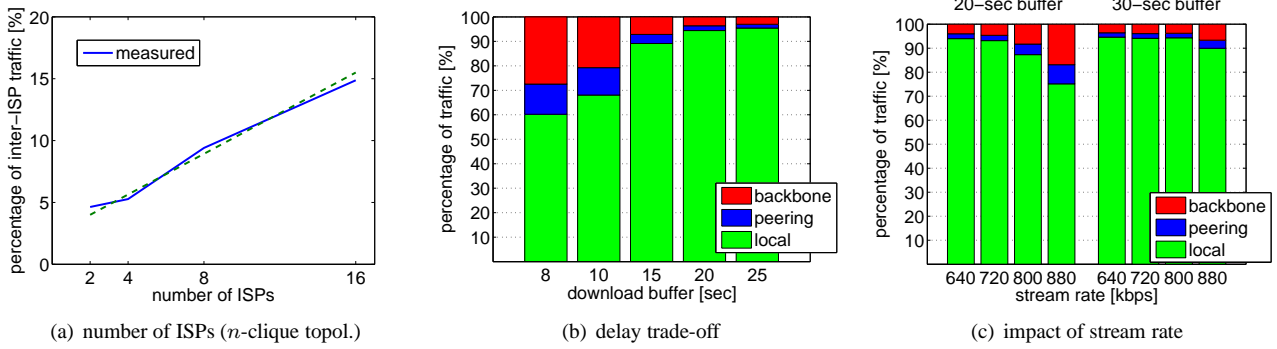


Figure 4. (a) inter-ISP traffic increases linearly with the number of ISPs. (b) increasing the download window size reduces inter-domain traffic. (c) higher stream rates increase inter-ISP traffic due to higher diffusion delays.

tice that 10 seconds corresponds to the mid-buffer position (the download buffer is 20 seconds). Thus, our dynamic choke mechanism, driven by mid-buffer misses, is pushing the diffusion delay towards the mid-buffer value. As we will see in Section 5.4, this delay increase can be exploited to further reduce the amount of inter-ISP traffic.

Finally, we compare the 1-peer and full schemes in the presence of a network partition. In this case, we use only the 2+2 unbalanced scenario, where slow clients in ISP A2 download chunks from fast ones in ISP A1 (the same holds for B2/B1). After letting the overlay stabilize, we kill all the clients in A1. From A2’s perspective, this is equivalent to a network partition which disconnects all peers in A2 from their uploaders in A1. Thus, all peers in A2 must quickly find new uploaders in B1 and B2. Figure 3(b) shows the effect of a partition occurring after $t = 6$ minutes. The bottom curve shows the average miss ratio over all peers in the overlay. The chunk misses in the first two minutes are due to all nodes joining the overlay simultaneously (i.e., a flashcrowd effect). In the full scheme, peers in ISP A2 react quickly to the network partition by unchoking their secondary peerset, which already contains, in average, several peers in ISPs B1 and B2. The few deadline misses we observe are due to chunks which were injected into ISP A1 by the source right before the partition, and are therefore lost before they are propagated to other ISPs. The 1-peer scheme shows a much slower reaction time. Peers in ISP A2 are not able to find new uploaders quickly enough, and their single secondary peer is not sufficient to download all necessary chunks. As a result, they experience deadline misses during several minutes until the overlay converges to its new configuration.

5.3. Impact of number of nodes and ISPs

We now measure the effect of the overlay size on non-local traffic. We use the 6+6 topology, which contains both

large and small ISPs. Figure 3(c) shows that the percentage of peering traffic decreases for larger overlays. In fact, increasing the number of peers means that small ISPs contain more nodes, thus generating more intra-ISP and less peering traffic. This suggests that our scheme will be most efficient when streaming popular channels, i.e., watched by a large number of users. Moreover, the overall system efficiency should also be high provided most traffic is generated by a small number of popular channels. This is the case in current systems, as recent studies show that channel popularity follows a Zipf distribution [25, 26].

We also evaluate the scalability of our solution in terms of the number of ISPs. We use the n -clique topology, a 30-second download buffer, and a homogeneous capacity distribution. Notice that in this scenario, each of the n ISPs must receive at least one copy of the stream to be able to diffuse it internally, so inter-ISP traffic is $\Omega(n)$. Figure 4(a) shows the inter-ISP traffic for 320 clients distributed uniformly among an increasing number of ISPs. We observe that non-local traffic grows linearly with n . Given the linear lower bound, this suggests that our system achieves good scalability.

5.4. Impact of buffer size and stream rate

In the following experiment we evaluate the impact of the download buffer size. We use the 2+2 balanced homogeneous scenario, and measure the local and peering traffic after overlay stabilization for different download buffer sizes. Figure 4(b) shows that the fraction of local traffic increases for larger download windows. Intuitively, increasing the download buffer also increases the mid-buffer position, allowing more chunks to propagate through primary edges (i.e., low-cost links) instead of triggering secondary unchokes. This behavior shows a trade-off between the efficiency of the system (in terms of costly traffic) and the

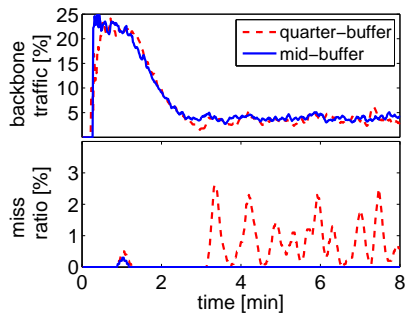


Figure 5. unchoke based on quarter-buffer misses are too close to the deadline, producing periodic chunk misses.

diffusion delay. Thus, when streaming delay-critical content (e.g., live sport events), the system could employ short delays at the cost of more inter-ISP traffic. Conversely, pre-recorded TV shows and other delay-tolerant content may be streamed at lower network costs by increasing the download buffer. Note that if the download buffer is too small, all peers will completely unchoke their secondary peersets. In our experiments, this occurs for a 8-second buffer³. Although the advantages of the dynamic secondary unchoke mechanism are lost with such a short buffer, the system still generates much less costly traffic than an ISP-agnostic scheme thanks to the ISP-aware overlay construction (cf. Figures 4(b) and 2(b)).

Another factor that influences the efficiency of the system is the ratio between the average upload capacity and the stream rate. Figure 4(c) shows that inter-ISP traffic increases for higher stream rates. In fact, a higher stream rate means more data is sent through the clients' uplinks, increasing the chunk transmission time and the average diffusion delay. This, in turn, produces more mid-buffer misses, resulting in more frequent secondary unchokes. As shown in Figure 4(c), this effect can be mitigated by increasing the download buffer size, which reduces inter-ISP traffic thanks to the delay/efficiency trade-off discussed before.

5.5. Mid-buffer vs. quarter-buffer miss

We now compare the use of two different early starvation signals (ESS), mid-buffer miss and quarter-buffer miss. The latter is generated if a chunk is still missing when only 25% of the total download buffer time is left before its deadline. We use the 2+2 balanced homogeneous scenario. The download buffer is 20 seconds, as in the previous experiments.

³A smaller download buffer will result in deadline misses. As shown in Figure 3(a), even an agnostic overlay needs around 7 seconds to deliver all chunks.

Figure 5 shows that using quarter-buffer misses results in periodic deadline misses across the entire overlay, which does not occur when using mid-buffer misses. The reason is that unchoke secondary sources only 5 seconds before the deadline is too late to avoid starvation. A large number of chunks may be missing by the time the ESS is generated, in which case secondary sources will not have enough time to upload all missing chunks before their deadline. Such oscillations could be reduced by adjusting the multiplicative factors α and β . However, this can result in a slower adaptation, decreasing the reactivity of the system to changing network conditions.

5.6. Churn

In this experiment we evaluate our design's resistance to churn. Our scheme must ensure that primary peersets are quickly reconfigured upon connections and disconnections, maintaining an overlay configuration close to the optimal. Furthermore, churn must not cause chunk misses which would trigger excessive secondary unchokes, decreasing the efficiency of the system. We simulate churn by killing and restarting a randomly chosen node every 5 seconds. This ensures that the total overlay size remains constant. The mean peer session time is $5 * 320 = 1600$ seconds, which corresponds roughly to the value measured on a real system [13] (1500 seconds). We use the two 2+2 unbalanced scenarios, as these converge more slowly than the balanced ones, and thus are more sensitive to churn.

Figure 6(a) shows that under churn the overlay converges to a slightly costlier configuration with respect to the static case. The difference is more evident in the heterogeneous scenario, as the overlay converges more slowly, and thus the impact of churn is higher. In the heterogeneous case, churn also produces a slight increase in backbone traffic. This is caused by the suboptimal overlay configuration, as our traces show that churn does not affect the dynamic secondary unchoke mechanism. These experiments suggest that a faster overlay rewiring algorithm could make the overlay converge to a lower cost configuration under churn, reducing backbone traffic.

5.7. Planetlab

Finally, we evaluate our prototype's behavior on Planetlab. Since Planetlab machines are spread across a large number of Autonomous Systems (AS), we select 160 machines and assign them a logical AS number according to our synthetic topology. In this experiment we run 160 peers and use the two 2+2 homogeneous scenarios. The goal of our Planetlab experiments is to verify that our prototype behaves as in our emulation-based experiments.

The results are shown in Figures 6(b) and 6(c). First, we

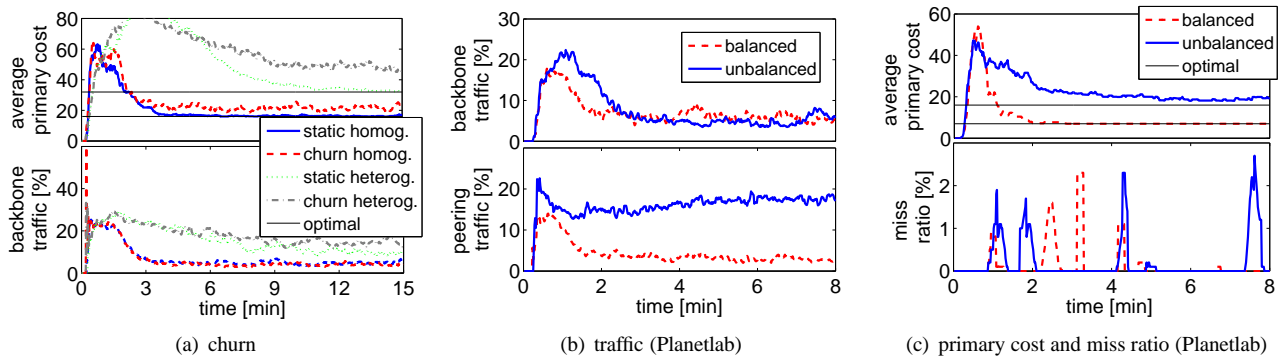


Figure 6. (a) the overlay converges to a slightly costlier configuration under churn. (b,c) Planetlab experiments (160 peers) show sporadic chunk misses, due to overloaded machines. Traffic and overlay configuration are almost identical as with emulation.

observe that the backbone and peering traffic values are stable over time, and are very close to those obtained in emulation experiments of Figure 2(b). The overlay convergence is also comparable to that of Figure 2(a). The only difference we observe is the presence of chunk misses, as shown in Figure 6(c). Our traces show that the peaks correspond to a large number of peers experiencing a low miss ratio (1-3%), i.e., no peer experiences high miss ratios. We conclude that these peaks correspond to chunks whose diffusion has been delayed when only a few copies existed in the overlay. This behavior, not observed in emulation experiments, is most likely caused by a temporary CPU or network starvation experienced by the peer which was relaying such chunks. Given that the miss ratio is low, missing chunks could be reconstructed by encoding the stream with a small amount of redundancy, e.g., using erasure codes [27]. More importantly, the absence of perturbations in the backbone and peering traffic shows that our scheme is robust against such isolated chunk misses.

6. Conclusions and future work

In this paper we have presented a novel approach for P2P streaming that dramatically reduces inter-ISP traffic with no impact on quality. We have shown both analytically and experimentally that our scheme achieves highly efficient P2P live streaming, while providing good reactivity to churn and sudden changes in the network. Moreover, our solution can be tuned to achieve different cost/delay trade-offs, and provides good scalability in terms of number of peers and ISPs.

We plan to further refine our solution by studying more efficient secondary unchokes. In particular, more sophisticated early starvation signals, such as analyzing the instantaneous delay distribution, could be employed to avoid the delay overhead incurred by the mid-buffer miss mechanism.

We will also investigate the use of rate-limiting on primary edges, which may further increase efficiency in scenarios where primary edges are created between two ISPs due to a limited upload capacity within one of them. Finally, we will look into faster peer discovery algorithms that increase the overlay's convergence speed, and thus should reduce inter-ISP traffic under churn.

References

- [1] CacheLogic: The True Picture of Peer-To-Peer. http://web.archive.org/web/200406-re_/http://www.cachelogic.com/research/slide1.php
- [2] PPLive, <http://www.pplive.com>
- [3] G. Huang, Experiences with PPLive. Keynote at *ACM. SIGCOMM P2P-TV*, Aug. 2007.
- [4] TVants, <http://tvants.en.softonic.com>
- [5] SopCast, <http://www.sopcast.org>
- [6] B. Cohen. Incentives Build Robustness in BitTorrent. In *Proc. of P2PECON*, 2003.
- [7] G. Shen, Y. Wang, Y. Xiong, B. Y. Zhao, and Z.-L. Zhang. HPTP: Relieving the tension between ISPs and P2P. In *Proc of IPTPS*, Bellevue, WA, Feb. 2007.
- [8] T. Karagiannis, P. Rodriguez, and K. Papagiannaki. Should internet service providers fear peer-assisted content distribution? In *Proc. of the Internet Measurement Conference*, Berkeley, CA, Oct. 2005.
- [9] V. Aggarwal, A. Feldmann, and C. Scheideler. Can ISPs and P2P systems cooperate for improved performance? *ACM CCR*, July 2007.
- [10] H. Xie, Y. R. Yang, A. Krishnamurthy, Y. Liu, and A. Silberschatz. P4P: Provider Portal for Applications. In *Proc. of SIGCOMM*, 2008.
- [11] D.R. Choffnes and F. E. Bustamante. Taming the Torrent: A Practical Approach to Reducing Cross-ISP Traffic in Peer-to-peer Systems. In *Proc. of SIGCOMM*, 2008.

- [12] P. Maymounkov and D. Mazieres. Kademia: A Peer-to-peer information system based on the XOR metric. In *Proc. of International Workshop on Peer-to-Peer Systems (IPTPS)*, 2002.
- [13] M. Zhang, Q. Zhang, L. Sun, and S. Yang, Understanding the Power of Pull-based Streaming Protocol: Can We Do Better? In *IEEE JSAC, special issue on Advances in Peer-to-peer Streaming Systems*, 2007.
- [14] A. Vahdat, K. Yocum, K. Walsh, P. Mahadevan, D. Kotic, J. Chase, and D. Becker. Scalability and accuracy in a large-scale network emulator. In *Proc. of OSDI*, 2002.
- [15] PlanetLab. <http://www.planet-lab.org>
- [16] C. Liang, Y. Guo, and Y. Liu. Is Random Scheduling Sufficient in P2P Video Streaming? In *Proc. of ICDCS*, 2008.
- [17] T. Bonald, L. Massoulié, F. Mathieu, D. Perino, A. Twigg. Epidemic Live Streaming: Optimal Performance Trade-Offs. In *Proc. of SIGMETRICS*, 2008.
- [18] F. Picconi and L. Massoulié. Is there a future for mesh-based live video streaming? In *Proc. of P2P*, 2008.
- [19] MATLAB. <http://www.mathworks.com>
- [20] R. Oliveira, D. Pei, W. Willinger, B. Zhang, and L. Zhang. In Search of the elusive Ground Truth: The Internet's AS-level Connectivity Structure. In *Proc. of SIGMETRICS*, 2008.
- [21] X. Dimitropoulos and et al. AS relationships: Inference and validation. In *ACM SIGCOMM Computer Communication Review (CCR06)*, 37(1):2940, January 2007.
- [22] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller. Equations of State Calculations by Fast Computing Machines. In *Journal of Chemical Physics*, 21(6):1087-1092, 1953.
- [23] V. Ribeiro, R. Riedi, R. Baraniuk, J. Navratil, and L. Cottrell. pathChirp: Efficient Available Bandwidth Estimation for Network Paths. In *Passive and Active Measurement Workshop*, 2003.
- [24] <http://www.journaldunet.com/ebusiness/telecoms-fai/analyse/070831-parts-de-marche-haut-debit/2.shtml>
- [25] X. Hei, C. Liang, J. Liang, Y. Liu, and K. W. Ross. A Measurement Study of a Large-Scale P2P IPTV System. In *IEEE Trans. on Multimedia*, 9(8):1672-1687, 2007.
- [26] M. Cha, P. Rodriguez, S. Moon, and J. Crowcroft. On Next-Generation Telco-Managed P2P TV Architectures. In *Proc. of International Workshop on Peer-To-Peer Systems (IPTPS)*, February 2008.
- [27] R. Blahut. Theory and Practice of Error Control Codes. Addison Wesley, MA, 1994
- [28] E.K. Lua, X. Zhou, J. Crowcroft, and P.V. Mieghem. Scalable multicasting with network-aware geometric overlay. In *Computer Communications*, 31:464-488, 2008.
- [29] L. Massoulié, A.-M. Kermarrec, and A.J. Ganesh. Network awareness and failure resilience in self-organizing overlay networks. In *Proc. of Reliable Distributed Systems*, 2003.
- [30] B. Bollobás. Random graphs, 2nd Edition. Cambridge University Press, 2000.
- [31] D. Dubhashi and D. Ranjan. "Balls and Bins: A Study in Negative Dependence". BRICS report RS-96-25, 1996.

Appendix

We now provide the proof of Proposition 1 in Section 3.4. Below we note $b = d_{i,max}$ the stream capacity (expressed in units of nominal edge capacity), also assumed to coincide with the uplink bandwidth of all peers.

We first establish that, for a suitable level of choking, the cost resulting of full use of all edges is the one announced in the proposition.

We consider the same bandwidth limit of ϵb over every secondary peer set for all peers, where ϵ is a small parameter, and will assume that each secondary edge is used with capacity ϵ , while each primary edge is used with capacity $1 - \epsilon$, thus meeting the uplink bandwidth limitation of peers.

The parameter ϵ should be sufficiently large to ensure that for each ISP J , data from other ISPs can enter it at a rate no less than $b - \lambda_{0J}$. For large number of peers n , the number of secondary edges leading to ISP J from other ISPs is with high probability $(1 + o(1))bp(J)(n - n(J))$. Indeed, each edge emanating from all the other ISPs has an individual probability of $p(J) = n(J)/n$ of pointing towards ISP J . These events are negatively correlated, and there are $b(n - n(J))$ such edges, hence the result⁴. Thus, to ensure feasibility, one should impose

$$\epsilon \geq (1 + o(1)) \sup_{J \in \mathcal{J}} \frac{b - \lambda_{0J}}{bp(J)(n - n(J))}. \quad (3)$$

When each secondary edge is used with capacity ϵ , and each primary edge is used with capacity $(1 - \epsilon)$, the overall cost of using the overlay is given by:

$$c = \sum_{J \in \mathcal{J}} \frac{\underline{c}bn(J) \{(1 - \epsilon) + \epsilon(p(J) + o(1))\}}{+\bar{c}\epsilon n(J)b(1 - p(J))(1 + o(1))}. \quad (4)$$

Consider first the symmetric case. The right-hand side of Equation (3) reads:

$$\epsilon_{sym} := (1 + o(1)) \frac{b(1 - 1/|\mathcal{J}|)}{(b/|\mathcal{J}|)n(1 - 1/|\mathcal{J}|)} = (1 + o(1)) \frac{|\mathcal{J}|}{n}. \quad (5)$$

Plugged into Equation (4), this yields the following expression for the "symmetric cost" c_{sym} :

$$\begin{aligned} c_{sym} &:= \frac{\underline{c}bn}{+\bar{c}\epsilon_{sym}nb(1 - 1/|\mathcal{J}|)(1 + o(1))} \left\{ (1 - \epsilon_{sym}) + \epsilon_{sym} \left(\frac{1}{|\mathcal{J}|} + o(1) \right) \right\} \\ &= c^* + o(\bar{c}b|\mathcal{J}|) \end{aligned}$$

as announced.

⁴A more detailed argument is as follows. The total random number of edges is, because of negative correlations, less variable than the Binomial random variable with parameters $(b(n - n(J)), p(J))$ [31]. That is to say, Chernoff bounds for this Binomial random variable also apply to the considered number of edges. Thus this number is almost surely equal to $(1 + o(1))p(J)b(n - n(J))$.

In the asymmetric case, consistent with (3), we let

$$\begin{aligned}\epsilon_{asym} &= (1 + o(1)) \sup_{J \in \mathcal{J}} \frac{1}{p(J)(n-n(J))} \\ &= O(1/n).\end{aligned}$$

Plugged into (4), this yields the expression for the ‘‘asymmetric cost’’ c_{asym} :

$$\begin{aligned}c_{asym} &= \sum_{J \in \mathcal{J}} \bar{c} b n(J) (1 + O(1/n)) \\ &\quad + \bar{c} O(1/n) n(J) b (1 - p(J)) (1 + o(1)) \\ &= c^* + O(\bar{c} b |\mathcal{J}|)\end{aligned}$$

as announced.

We now show that with high probability, feasibility holds. We take an asymptotic viewpoint, assuming that the collection of ISP’s, \mathcal{J} , is fixed, but that the sizes $n(J)$ are given by $n(J) = p(J)n$ for some fixed parameter $p(J)$, and where the number of peers n is large.

Under these conditions, we have the following.

Proposition 2. *For fixed integer b , let $G(n, b)$ be a random graph on n nodes, with directed edges, in-bound and out-bound degree of b for all nodes, no loops (but possibly multiple edges), that is uniformly distributed over all graphs with such properties.*

Then, assuming $b \geq 2$, the following property holds with high probability (i.e. with probability $1 - o(1)$) as n goes to infinity.

For any subset S of m nodes with $1 \leq m \leq n - 1$, the number of edges directed from S to its complement is at least b .

Let us show how this result implies feasibility of streaming in our present context. Feasibility holds under the following condition: for any subset S of nodes (including the source), the sum of capacities of edges directed from S to its complement must be at least the stream rate, b . Let us distinguish two cases.

Case 1: S consists precisely in the union of the receivers in a collection of ISPs. Then, by our choice of ϵ , the number of secondary edges leading from S into its complement is at least b/ϵ , and hence the corresponding capacity is at least b .

Case 2: For some ISP J , the intersection of S with J ’s receiver set is non-empty and a strict subset of this receiver set. Then, by the above proposition applied to the graph of primary edges of receivers within J , there are at least b primary edges leaving the intersection of S with J and reaching other receivers of J . This accounts for a capacity of at least $b(1 - \epsilon)$. Similarly, the above proposition applied to the global graph of secondary edges now implies that there are at least b secondary edges leading from S to its complement. This accounts for a capacity of at least ϵb . Summing, one obtains a total capacity of at least b , as required.

Proof of Proposition 2:

We proceed as follows to generate the random graph $G(n, b)$. Consider a list of nb entries, corresponding to the edges leaving nodes in $G(n, b)$, and also to the edges entering nodes in $G(n, b)$. The graph $G(n, b)$ corresponds to a random uniform permutation of those nb entries, conditioned on the fact that this creates no loops.

We shall denote by \mathbf{P} the probability corresponding to the uniform permutation, without this conditioning. We need to bound by $o(1)$ the following quantity:

$$\pi := \mathbf{P}(\text{there is some set } S \text{ such that } E(S, \bar{S}) < b | \text{no loop}),$$

where $E(S, \bar{S})$ denotes the number of edges directed from S to its complement \bar{S} , and where one considers only sets S with cardinality between 1 and $n - 1$.

To this end, write

$$\begin{aligned}\pi &\leq \frac{1}{\mathbf{P}(\text{no loop})} \sum_{S: 0 < |S| < n} \mathbf{P}(E(S, \bar{S}) < b) \\ &= \frac{1}{\mathbf{P}(\text{no loop})} \sum_{m=2}^{n-2} \binom{n}{m} \sum_{\ell=0}^{b-1} \mathbf{P}(E(S_m, \bar{S}_m) = \ell),\end{aligned}\tag{6}$$

where S_m is an arbitrary set of m nodes. The restriction of the sum to indices m satisfying $2 \leq m \leq n - 2$ is justified by the fact that, conditionally on the absence of loops, the number of edges leaving sets of size 1 is always b ; similarly, the number of edges leaving sets of size $n - 1$ is always b , for otherwise the complementary set, of size 1, would be a node with a loop.

Let us evaluate the latter probability in (6).

$$\begin{aligned}\mathbf{P}(E(S_m, \bar{S}_m) = \ell) &= \frac{\binom{mb}{\ell} \binom{(n-m)b}{\ell} (\ell!)^2 (mb-\ell)! ((n-m)b-\ell)!}{nb!} \\ &= \frac{\binom{mb}{\ell} \binom{(n-m)b}{\ell}}{\binom{nb}{mb}}.\end{aligned}$$

Indeed, the first equality is justified as follows. The numerator counts the number of permutations that map exactly ℓ of the first mb entries to the last $(n - m)b$ entries. The second equality follows by straightforward simplifications.

Plugging this expression into (6), we obtain:

$$\pi \leq \frac{1}{\mathbf{P}(\text{no loop})} \sum_{m=2}^{n-2} \binom{n}{m} \sum_{\ell=0}^{b-1} \frac{\binom{mb}{\ell} \binom{(n-m)b}{\ell}}{\binom{nb}{mb}}$$

Denote by $f(m)$ the m -th term in the above summation. One then has

$$\begin{aligned}f(m) &= \binom{n}{m} \sum_{\ell=0}^{b-1} \frac{\binom{mb}{\ell} \binom{(n-m)b}{\ell}}{\binom{nb}{mb}} \\ &\leq \binom{n}{m} b \frac{(mb)^{b-1} ((n-m)b)^{b-1}}{\binom{nb}{mb}} \\ &\leq C e^{-(b-1)(m \log(n/m) + (n-m) \log(n/(n-m)))} \times \dots \\ &\quad \dots e^{(b-1)(\log(n-m) + \log(m))}\end{aligned}$$

where we used Stirling’s formula between the second and third line, and C is a constant that depends only on b .

It can now be shown from the latter bound on $f(m)$ that, when $b \geq 2$, the sum $\sum_{m=2}^{n-2} f(m)$ is $o(1)$ as $n \rightarrow \infty$.

To conclude, it remains to show that $\mathbf{P}(\text{no loop})$ is bounded away from zero. This can be achieved by first noting that, for any node, under the uniform permutation probability \mathbf{P} , it has a loop with probability $(b/n)(1+o(1))$, and by then using Poisson approximation arguments to show that the number of loops, under \mathbf{P} , follows asymptotically a Poisson distribution (see e.g. Bollobás [30]). This then entails that

$$\lim_{n \rightarrow \infty} \mathbf{P}(\text{no loop}) = e^{-b},$$

from which the desired result follows.