By how far have recent churn-reduction techniques improved quality of experience in peer-to-peer video streaming environments?

Martin Symons

April 30, 2024

Abstract

abstract !!!

1 Introduction

Within the past ten years, video streaming traffic across the internet has exploded. Traditional client-server architectures places a large load on the small portion of nodes controlled by the streaming host, and lead to dramatic infrastructure costs. To combat this, peer-to-peer systems have been proposed that spread usage across the network's collective resources. Starting with Coolstreaming in 2004, these networks became defacto for IPTV streaming, especially pirated streams. With their utility proven, research accelerated, in pace and complexity. Industry, however, did not pick up. Since the shutdown of New Coolstreaming and PPLive in the late 2000's, these newer solutions have seen little real-world usage.

As such, the actual impact of current research on network performance is vague. Still, research since has suggested that churn has a substantial and exponential impact on client quality-of-service (QoS) beyond that which was considered in these original models. This paper investigates several marked improvements over the best-documented benchmark architecture, New Coolstreaming, with particular focus on methods to reduce churn throughout the network. wordsw words words

2 Literature Review

2.1 Peer-to-Peer Streaming Fundamentals

In the late 1990's the traditional client-server architecture began to crack, as computer capabilities began to outpace bandwidth growth across the internet. Small clusters of server nodes were expected to take the burden, whilst ample client resources waited idle. BitTorrent allows a swarm of peers to collaborate in file transmission, maximizing global utilization without overwhelming any single node. Peers instead connect to a tracker or query a global DHT, usually *Mainline DHT*, to gather a list of participatory nodes. Targetting the rarest blocks first, data chunks within the swarm are piped directly over UDP node-to-node. As more peers join the network, the overall bandwidth increases, with download speeds following suit.

This approach exploded onto the internet, claiming 35% of all upstream traffic (**archiveCacheLogicResearch**). It remained in the number one spot for total traffic until recently - finally dethroned in 2024 by video streaming.

In a file-share network QoS is taken mostly with upload speed, and by extension bandwidth utilization. Any intermediate action from loading a *.torrent* to completion is irrelevant. Video streaming, in contrast, introduces a large number of real-time requirements essential for a good user experience.

Peer-to-peer networks pipe data directly between nodes through a network, contrasted with the traditional client-server architecture seen most often today. Compared to file-sharing applications, peer-to-peer video networks face several new complexities. In the former, QoS is ruled almost entirely by bandwidth utilization. Users care only that a desired file eventually reaches them; the journey it takes to completion is unimportant. probably list some examples Video livestreaming, however, introduces new complexities. playoutdeadlines startupdelay end-to-enddelay one proposed solution was ip multicasting. it was . fucked. s owe don't use it and instead use the various better solutions

2.2 Performance Heuristics and Churn

explaining why we use/care about churn vs other characteristics

2.3 Overlay Structure

2.4 Peer Selection Strategies

2.5 Chunk Schedulers

2.6 Historic Implementations

We now consider historic implementations of peer-to-peer streaming systems in relation to the above characteristics. Of most relevance to our paper is DONet, commercially branded Coolstreaming (**1498486**). Considered the first of its breed, Coolstreaming Xie et al. 2007 **asdasd 1498486**

2.7 Implementation-Specific Improvements

3 Methodology

We aimed to pit three models against each other in various QoS tests to gather specific numbers on the improvements between each. Initial tests would be run on New Coolstreaming, widely considered the last popular IPTV P2P system and a good benchmark . We would then extend this with fitting modular improvements for further measurements. were particular targets, because . Finally, we expected to implement a monolithic model, , to compare the capacity of incremental improvements versus the design of a single, deeply-coupled architecture.

As a simulation environment, we chose OverSim. Its included churn mechanisms, quick-switching between simplified and realistic underlay models, and complete debugging suite eased the otherwise involved development cycle. Ejecting from OverSim to OMNET++ would also have been trivial, though was never necessary.

Statistics would be presented with the built-in OMNET++ visualization tools.

which?

we probably already explained this in the lit review

name some shit

blah

name

needs expansion. we could mention dates?? what

4 Implementation

We based our initial experiments on New Coolstreaming as described in B. Li et al. 2008. We quickly ran into problems - whilst the paper describes the stream manager and buffer map exchange in great detail, little time is given to the membership and partnership managers. The upkeep of the *mCache* with incoming peers is unspecified, as is most connection management action related to churning or failing nodes and some key equations to system function. We pushed forward and attempted to fill the blanks ourselves. The final result, whilst technically functional, invariably failed to meet playout across nodes and was in no way correspondent of New Coolstreaming's measured real-world performance.

The New Coolstreaming paper concludes its discussion on the problem modules stating *"these basic modules form the base for the initial Coolstreaming system,"* and that the New Coolstreaming *"has made significant changes in the design of other components."* We thus considered that these modules were holdovers from the older design, and that this statement implied New Coolstreaming must be built with DONet/Coolstreaming as groundwork. We thereby set about an implementation of this more primitive design.

The final DONet implementation completed following two weeks of work. This network was similarly not ideal, though the cause was mostly banal - New Coolstreaming strips the buffer, scheduler and related messaging completely, so we saw no need to optimize these components. More worryingly, the partnership manager collapsed quickly under even minimal churn, discussed later in section . Still, this constituted enough the groundwork needed SECTIONWHAT to continue.

Returning with wiser eyes, we found that our architecture did not align at all with the basic modules in New Coolstreaming. *How could this be?* As discussed later in section , DONet has clarity problems of its own when SECTION describing parts of other systems, and we noticed that the output of these components - M-number exchanging partners ready for video transmission - *did* align with the older model. We therefore treated this as a simple faulty description, and moved on to the design of the stream manager.

The well-specified stream manager came through without a hitch, but placed new constraints on the partnership manager that our already brittle implementation could not bear. We hence designed *Partnerlink*, a relationship algorithm reconciling the high-churn overlay with New Coolstreaming's low-churn subscription requirements and performance at scale. This new algorithm meshed well with the stream manager, and brought our implementation to a close.

The full development process took over a month. We were therefore not able to complete any further models or make any comparisons on QoS benefits.

5 Results

jsut offhandedly mentioning that Yes, It Workas Kind Of

6 Discussion

What went wrong? New Coolstreaming is not unique as an overlay; no features within should prove particularly challenging to implement. In this section, we explore the Coolstreaming family as a whole, and illuminate the many challenges faced in their implementation amiss in the papers themselves.

6.1 The Coolstreaming Family Tree

We have so far regarded Coolstreaming as a dyad of the mesh-pull DONet/Coolstreaming and hybrid-push-pull New Coolstreaming, proposed across two papers. The reality is not so simple. Coolstreaming is formed of two models, as described. However, they have been proposed under *four* different names across *four* papers:

- As *DONet* and *Coolstreaming*, in *CoolStreaming/DONet: a data-driven overlay network for peer-to-peer live media streaming* Zhang et al. 2005
- As *Coolstreaming*, in *Coolstreaming: Design, Theory, and Practice* Xie et al. 2007
- As *Coolstreaming+*, in *An Empirical Study of the Coolstreaming+ System* Bo Li et al. 2007
- As *"the new Coolstreaming,"* in *Inside the New Coolstreaming: Principles, Measurements and Performance Implications* B. Li et al. 2008.

Note that the name *Coolstreaming* is intended by this final paper, but *The New Coolstreaming* became the colloquial model name as a result of its title. Only the first paper contains the old model we have discussed as *DONet/Coolstreaming* - the others all specify *New Coolstreaming*, despite the name differences.

The name *Coolstreaming* legally refers both to the older *DONet* model and the newer *New Coolstreaming* model. The confusion that results is obvious. Kondo et al. 2014 describes the *SCAMP* membership protocol, the push-pull mechanism and the bootstrap node as part of the same model, despite *SCAMP* being specific to the mesh-pull *DONet.* Beraldi, Galiffa, and Alnuweiri 2010 makes much the same error. Lan et al. 2011 takes *DONet* as its key example of a buffer-map driven overlay, but ascribes it the synchronization method seen in *New Coolstreaming.* Further examples still can be found of correct prose, but with Xie et al. 2007 being cited in Zhang et al. 2005's place, or vice versa.

It is interesting to note that the papers themselves appear to have issues keeping the versions straight: B. Li et al. 2008 describes the stream manager and new mCache system as part of the *"initial Coolstreaming system"* and replaced in *"the new Coolstreaming system"*, despite all of these components being local to *New Coolstreaming* only - hence our initial hesitance to visit *DONet*.

This escalates considering the final three papers. Xie et al. 2007 is the canon definition of *New Coolstreaming*, alongside analysis of a real-world test period to determine convergence rates, start-up delay and other overlay-specific statistics. The other papers duplicate this and add further analysis: Bo Li et al. 2007 splits users into categories, identifying network traversal problems and their respective impact on contribution. B. Li et al. 2008 performs an additional simulation to identify ideal values for key system parameters.

This duplication means each paper opens with a definition of *New Coolstreaming*. These are not summaries - the *Coolstreaming+* specification is shortest yet still clocks in at two and a half pages. The majority of this text is word-for-word identical between papers, or at best reworded. The membership and partnership managers, though, have their specifications cut down completely, no longer parsing as a working system. For instance, connection management is resolved in Xie et al. 2007 by an off-handed mention of TCP - which includes a leaving and timeout mechanism, if specified. In the other papers, TCP is never mentioned; no other connection management is described. Mechanisms to fill the bootstrap node's *mCache* alongside one key function related to playout initialization are similarly constrained to this paper, despite these papers claiming to *"describe the architecture and components in the new Coolstreaming system"*.

Luckily, we appear to be the only researchers to fall into this trap. Resulting development time was certainly extended, but our criticisms of the paper and our final solution remain valid.

6.2 The Fully-Connected Network Problem

the difficulty in specifying a partnership protocol that can handle networks of M+1. this is no longer about it being impossible !! simply that this is non-trivial and should have been specified

6.3 Choosing New Neighbours

disabling a neighbour inequality to force the network to funciotn. fuck off

6.4 Requesting the Block

lack of clarity in playout index calculations. we already know this sucks

6.5 Brainslugging and Production Optimization

DONet defers to *SCAMP* for membership connection management. SCAMP, as established ,

establish this

6.6 What does Coolstreaming need to be?

a research network. nobody cares about this thing for absolute production performance anymore, we are far far beyond that point. it is more important to operate as a baseline.

6.7 Introducing ***coolstreaming-spiked***

our proposals for a better coolstreaming for the research community. unfucked scamp, the partnerlink specified partnership protocol, and some other things that we clearly do not have answers for [if we did, we'd have a working network!] but should be Open Research Questions

6.8 The ***Partnerlink*** algorithm

simple definition of partnerlink. remember we need this because of the research network's shape - we want to keep the switching algorithm, and this was already a problem within donet, now worsened in new coolstreaming.

7 Conclusion

holy fuck. this sucks

References

Beraldi, Roberto, Marco Galiffa, and Hussein Alnuweiri. 2010. W-coolstreaming a protocol for collaborative data streaming for wireless networks. In *2010 ieee 30th international conference on distributed computing systems workshops,* 221–226. https://doi.org/10.1109/ICDCSW.2010.78.

Kondo, Daishi, Yusuke Hirota, Akihiro Fujimoto, Hideki Tode, and Koso Murakami. 2014. P2p live streaming system for multi-view video with fast switching. In *2014 16th international telecommunications network strategy and planning symposium (networks),* 1–7. https://doi.org/10.1109/NETWKS.2014.6959253.

Lan, Shanzhen, Qi Zhang, Xinggong Zhang, and Zongming Guo. 2011. Dynamic asynchronous buffer management to improve data continuity in p2p live streaming. In *2011 3rd international conference on computer research and development,* 2:65–69. https://doi.org/10.1109/ICCRD.2011.5764085.

Li, B., S. Xie, Y. Qu, G. Y. Keung, C. Lin, J. Liu, and X. Zhang. 2008. Inside the new coolstreaming: principles, measurements and performance implications. In *Ieee infocom 2008 - the 27th conference on computer communications,* 1031–1039. https://doi.org/10.1109/INFOCOM.2008.157.

Li, Bo, Susu Xie, Gabriel Y. Keung, Jiangchuan Liu, Ion Stoica, Hui Zhang, and Xinyan Zhang. 2007. An empirical study of the coolstreaming+ system. *IEEE Journal on Selected Areas in Communications* 25 (9): 1627–1639. https://doi.org/10.1109/JSAC.2007.071203.

Xie, Susu, Bo Li, Gabriel Y. Keung, and Xinyan Zhang. 2007. Coolstreaming: design, theory, and practice. *IEEE Transactions on Multimedia* 9 (8): 1661–1671. https://doi.org/10.1109/TMM.2007.907469.

Zhang, Xinyan, Jiangchuan Liu, Bo Li, and Y.-S.P. Yum. 2005. Coolstreaming/donet: a data-driven overlay network for peer-to-peer live media streaming. In *Proceedings ieee 24th annual joint conference of the ieee computer and communications societies.* Vol. 3, 2102–2111 vol. 3. https://doi.org/10.1109/INFCOM.2005.1498486.