

# Improving Medium-Sized Media Clip Distribution Through Transparent Tail Synchronization

A. Striegel, D. Salyers, D. Moore, Y. Jiang, A. Blaich  
Department of Computer Science and Engineering  
University of Notre Dame  
Notre Dame, IN 46556  
Email: {striegel,dsalyers,dmoore7,yjiang3,ablaich}@nd.edu

**Abstract**—The emergence of popular video sharing sites such as YouTube has created a tremendous content shift towards timely, medium-sized media together with placing significant demands on the network. While techniques such as Application Layer Multicast and P2P streaming offer the potential to reduce the impact of general streaming content, the difficulty in imposing synchronization and the heavy asymmetry of clients nullify the majority of the respective benefits of the techniques. In this paper, we propose a method, transparent tail synchronization, that discovers latent opportunities for synchronization from the tail of the content to take advantage of efficient distribution techniques. Our approach maximizes savings at the content provider while operating in a straightforward and easily deployable manner. We describe how tail synchronized media distribution can offer savings across a wide range of asymmetry at the end clients.

## I. INTRODUCTION

With the proliferation of high-speed network access to the home user, the scale of content has gravitated towards richer media interactions. For traditional broadcast media from live sources, numerous solutions ranging from IP multicast to Application Layer Multicast (ALM) [1] to Peer to Peer (P2P) streaming [2] have emerged to efficiently deliver content. For each of these respective solutions, a critical characteristic of the content is the natural synchronization of users due to the live broadcast nature of the content. The memory-less property of live content (i.e. future clients joining do not need previous content) allows for a de facto synchronization that is simply not present in all media.

In contrast, the emergence of rich media sites such as YouTube introduce a significantly different type of transfer. This medium-scale media dwarfs the requirements of typical content-laden web pages but yet pales in comparison to an average full length movie (DVD, HD-DVD, BluRay, etc.). We posit that this “medium-scale” content introduces interesting properties that complicate distribution of such media over previously dominant content. To start, content often arrives in a reliable manner via TCP due to both the simplicity of using TCP as well as the ability of TCP to easily navigate firewall/NAT mechanisms. Furthermore, the magnitude of the available content choices in terms of both choices and size creates numerous issues regarding the ability to effectively remotely cache content. These constraints are only exacerbated by the expectation of near immediate playback, i.e. show me the rich clip now, not two hours from now.

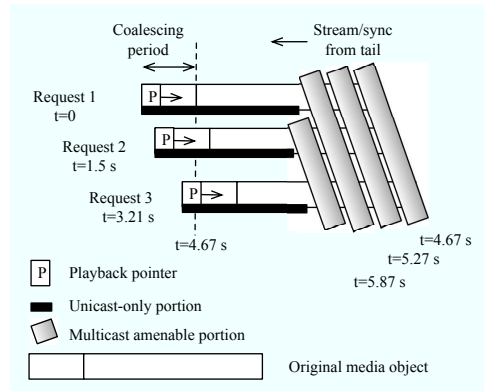


Fig. 1. Tail-synchronization of medium-sized media objects

The paper posits a relatively simple question: *is it possible to extract synchronization without imposing significant restrictions on the existing transfer mechanisms, i.e. is there latent potential for synchronization that can be exploited transparently?* To that end, we propose the concept of tail synchronization that virtually subdivides existing static media objects into zones for the purposes of coalescing, streaming, and completion. The root of tail synchronization emerges from how objects are synchronously streamed from the tail to minimize the impact of the unreliable aspects of streaming content on playback but conversely to maximize the amount of time available to stream content through the highly asymmetric pipes. Tail synchronization is not a replacement for distribution mechanisms but rather provides amenable content that can take advantage of such mechanisms (ALM, IP Multicast, Small Group Multicast [3], P2P, etc.). We believe tail synchronization is an extremely lightweight approach that can bring reasonable efficiency gains in an immediate timescale.

The remainder of this extended abstract is organized as follows. Section II briefly describes tail synchronization, preliminary results, and notable related work. Section III concludes by discussing open research questions.

## II. TAIL-SYNCHRONIZATION OVERVIEW

For each medium-sized media object offered by the content provider, the object itself is subdivided into three zones, namely the *coalescing zone*, the *streaming zone*, and the

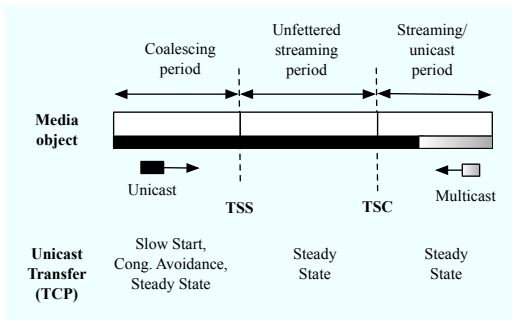


Fig. 2. Tail-synchronization of medium-sized media objects

*completion zone*. Figure 1 shows a conceptual view of tail synchronization. When the first request for the clip arrives, a virtual group of receivers is created. Each virtual group is tied to a specific object with a group closing trigger.

The first trigger (see Figure 2), *TSC* (Tail Sync Coalesce), represents the effective time during which subsequent requests can be coalesced into a single distribution grouping. Providing that sufficient space with a distribution grouping exists, clients requesting the same object will be grouped together for tail synchronized streaming. The boundary of coalescing zone is computed by multiplying the overall object size  $S$  times the *TSC* (ranges from 0 to 1). The first member of the group to cross the *TSC* barrier in its underlying TCP transfer of the media object triggers the virtual group to transition to the streaming zone and close itself to future membership additions.

Following the transition to the streaming zone, content is synchronously streamed from the tail of the media object to each of the virtual group members. The transport of the content may be streamed using traditional IP multicast, Automatic Multicast Tunneling (AMT), Application Layer Multicast (ALM) [1], or TCP overlays. Notably, the TCP transfers coming from the front of the object still transfer to satisfy playback timeliness constraints. Streaming from tail offers the opportunity for synchronization in addition to offering the ability to mask security, network, or capability errors from said streaming. Streaming of the content continues in earnest to all virtual receivers until one member passes the *TSS* (Tail Sync Stream) trigger of its TCP transfer.

Beyond the *TSS* trigger, an ambiguous state is reached whereby the synchronized transfer is proceeding slowly from the rear but yet the TCP transfer is coming from the front. To avoid overlap, the TCP transfer is given deference when the tail synchronized transfer approaches within a reasonable distance. Cleanup of missing data is taken care of via TCP and partial transfers. The net result (depending on asymmetry) is a significant opportunity for efficiency improvements.

#### A. Related Work

Due to space constraints, we comment on a limited subset of related work. First, tail synchronization shares a common lineage amongst the existing literature on video on demand and parallel video streaming [4]. In contrast to this previous work, tail synchronization focuses on the extraction of

synchronization rather than the ‘how’ of the transport or the distribution of resources (number of hosts, data striping, etc.). Second, tail synchronization is distinct from P2P media streaming solutions, ex. SplitStream [2], [5], in that the targeted content does not have forced synchronization but yet has timeliness constraints for playback. Finally, we note that tail synchronization does not preclude the use of intelligent digital fountain/coding-based schemes [6], [7] but rather offers an interesting mechanism for introducing synchronization.

### III. ON-GOING/FUTURE WORK

As noted earlier, this extended abstract represents a work in progress. Key research questions to be addressed as part of the research includes:

- *Flash functionality*: Is it possible to embed tail synchronization entirely in Flash despite support for only TCP transfers and a limited threading model? Is Ajax a better model for deployment?
- *Firewall/security*: How can one quickly isolate firewall issues in an increasingly NAT’d client environment? Will IPv6 worsen or simplify the usage of tail synchronization?
- *Client capability assessment*: How can one quickly assess client-side capabilities with regards to downstream streaming? How does one react to changes in client dynamics (background downloads, mobile nodes, etc.)?

The tail synchronization is a subset of our on-going Scale-Box research project. Additional information can be found at the NetScale wiki<sup>1</sup>.

### ACKNOWLEDGEMENTS

This work was supported in part by the National Science Foundation through the grant CNS03-47392.

### REFERENCES

- [1] Y. Chu, S. G. Rao, S. Seshan, and H. Zhang, “A case for end system multicast,” *IEEE Journal on Selected Areas in Communication (JSAC), Special Issue on Networking Support for Multicast*, Oct. 2002.
- [2] V. Padmanabhan, P. Chou, and H. Wang, “Resilient peer-to-peer streaming,” in *Proc. of IEEE ICNP*, 2003.
- [3] R. Boivie, “A new multicast scheme for small groups,” *IBM Research Report RC21512*, June 1999.
- [4] D. Ghose and H. Kim, “Scheduling video streams in video-on-demand systems: A survey,” *Multimedia Tools and Applications*, vol. 11, no. 2, pp. 167–195, June 2000.
- [5] M. Castro, P. Druschel, A. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, “Splitstream: High-bandwidth multicast in a cooperative environment,” in *Proc. of SOSP’03*, Lake Bolton, New York, Oct. 2003.
- [6] J. Byers, M. Luby, and M. Mitzenmacher, “A digital fountain approach to the reliable distribution of bulk data,” *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 8, pp. 1528–1540, Oct. 2002.
- [7] S. Rost, J. Byers, and A. Bestavros, “The cyclone server architecture: Streamlining delivery of popular content,” in *International Workshop on Web Caching and Content Distribution*, Boston, MA, June 2001.

<sup>1</sup>NetScale.cse.nd.edu