Signal Processing: *Image Communication* 27 (2012) 457–469

Contents lists available at SciVerse ScienceDirect

Signal Processing: *Image Communication*

journal homepage: www.elsevier.com/locate/image

Peer-to-peer streaming in heterogeneous environments

Remo Meier*, Roger Wattenhofer

Swiss Federal Institute of Technology, Distributed Computing Group, 8092 Zurich, Switzerland

ARTICLE INFO

Available online 21 February 2012

Keywords:
Peer-to-peer
Streaming
Scalable coding
Erasure coding

ABSTRACT

Peer-to-peer overlay networks are comprised of different kinds of devices, from mobile phones to high-definition televisions. They differ in size, computational power, and Internet access. The design of any peer-to-peer system has to account for such heterogeneous environments. For example, in the context of content delivery systems, the content must be delivered reliably, on time, and in a format suitable for each peer.

This work addresses the heterogeneity and reliability of peers in peer-to-peer streaming applications. It applies lessons learned from distributed hash tables (DHTs) by adopting a prefix-based overlay structure. The flexibility of its neighbor selection policy is exploited to make use of scalable coding and erasure coding schemes, bringing different kinds of peers together in a single overlay network. Thereby, each peer can select the appropriate number of scalable coding layers to obtain content in a suitable format. The prefix-based nature further allows efficient content distribution with low-delay, simple maintenance, strong connectivity, and quick adaption to changing conditions; making the proposed algorithms desirable for real-world use, for both peer-to-peer live and on-demand streaming.

1. Introduction

Peer-to-peer technology is an appealing paradigm for the distribution of audio and video content. It allows utilizing the resources of participating peers to overcome the shortcomings of more centralized, server-based approaches. More recently, there is a trend toward applications that deliver content in real-time, such as peer-to-peer streaming applications. At the same time, we witness a variety of devices from mobile phones and tablet computers up to high-definition three-dimensional televisions gaining access to the Internet. The design of any peer-to-peer system has to consider this heterogeneous nature of peers.

For file-sharing applications, the capabilities of peers only influence the average download time. In contrast, streaming applications face the challenge of delivering content with strict a playback deadline; content not delivered on time is of no use to a peer, and is discarded. This makes the design of any peer-to-peer streaming system more intricate. It is practically impossible to distribute content encoded in a single format, i.e., with a given resolution and bitrate. Peers may lack the bandwidth to sustain a stream. Weaker peers may be unable to process the incoming content. Action sequences in a video stream can lead to bursts of packets. Mobile devices lack the display to output high definition content. And changing network conditions may abruptly alter the situation of individual peers.

In this work we present novel techniques to cope with heterogeneous peers. A structured yet flexible overlay network is able to accommodate arbitrary heterogeneous sets of peers, and allows the delivery of both live and on-demand streams. The overlay employs a prefixed-based routing policy, similar to distributed hash tables, to gain desirable properties, such as an efficient distribution with low delay, robustness to churn, and guaranteed connectivity among all

* Corresponding author.
E-mail addresses: remmeier@tik.ee.ethz.ch (R. Meier), wattenhofer@tik.ee.ethz.ch (R. Wattenhofer).

0923-5965/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.image.2012.02.008

peers with a logarithmic overlay diameter. Scalable audio and video coding schemes (whereas content is partitioned into multiple coding layers) and erasure coding schemes complement the overlay structure to address the varying capabilities and reliability of peers. In contrast to other work, a *single* connected prefix-based overlay network is sufficient to distribute all coding layers. This eases the implementation, provides better robustness, and allows peers to more quickly adapt to changing network conditions.

The subsequent Section 2 reviews techniques to approach the heterogeneous nature of peers and studies related literature. Scalable coding schemes are presented in Section 3. Section 4 presents our overlay structure and content distribution mechanism. While the focus is on live streaming protocols, the used techniques are also applicable to on-demand and live/on-demand hybrid streaming protocols. Finally, Section 5 evaluates the proposed mechanisms and Section 6 concludes this paper.

2. Related work

Peer-to-peer live streaming protocols are mainly categorized according to the topology maintained among the peers or, equivalently, the neighbor selection algorithms the peers employ. Simple multicast systems are based on overlay *trees* [3,7,29]. While tree topologies are conceptually simple, there are rather serious drawbacks which render such systems inefficient. For example, resources are wasted as the leaves of such a tree do not contribute anything to the system, while inner nodes having two children need to upload at twice the bitrate of the stream. The fragile tree structure is not resistant to peer failures or churn, and a peer is limited by its weakest predecessor in the tree. While maintaining several trees [2] improves the robustness of a system, each tree can break individually, and the overhead potentially increases as more trees have to be repaired continuously (and concurrently).

Since a rigidly structured overlay requires permanent maintenance, care has to be taken not to burden the individual peers. Therefore, unstructured overlays have been favored over structured overlays, and various protocols based on unstructured overlays have been proposed, e.g., *CoolStreaming/DONet* [31], *Chainsaw* [18] and *GridMedia* [14]. Typically in unstructured overlays, peers have to *notify* neighboring peers about available blocks of data, and peers that are interested in obtaining these blocks must explicitly *request* them before any data is exchanged, because there is no structure in the network that can be used to disseminate data. Unstructured overlays are considered more robust, whereas they have the disadvantage that notifying peers and subsequently requesting data blocks potentially results in long delays before any data is exchanged. If further optimizations are applied (such as favoring connections among near-by peers) there is the additional concern that unstructured overlays fall apart due to the lack of structure and the formation of clusters.

In the literature, there are four main techniques to fix the lack of guarantees and the heterogeneous nature of peers: stream switching, source coding, multiple description coding (MDC), and scalable coding. Each coding technique has its own merits.

Stream switching encodes the content at different bitrates. A peer chooses a bitrate for download based on its capabilities and available bandwidth. The peer may later switch between bitrates to adapt to changing network conditions. For video streams, switching takes place at regular key frames or dedicated switching frames [9]. There are a variety of different implementations. More recently, HTTP Live Streaming[1] (also referred to as HLS) has been submitted to the IETF for standardization and is gaining in popularity due to its simplistic HTTP-based design. Stream switching works well in server-based environments, but it faces challenges when applied in peer-to-peer systems. Streams with different bitrates are independent of one another. Accordingly, peer-to-peer systems have to partition the peers into groups based on the chosen bitrate and thereby may separate near-by peers. Switching between streams is prohibitively expensive; peers have to replace connections to neighboring peers and the contents of their buffers. Furthermore, stream switching is not resilient to missing data blocks. Peers would have to choose a sufficiently low quality to ensure a timely delivery of all blocks at all times.

Erasure codes, such as Reed–Solomon codes [21], LT codes [13], and Raptor codes [25], allow the generation of n coded blocks from k original data blocks with $n>k$, whereas any $k+\epsilon$ coded blocks with $\epsilon \geq 0$ allow the reconstruction of the original data blocks. For a perfect erasure coding scheme, it holds that $\epsilon = 0$. However, most schemes trade the optimal message complexity for a lower computational complexity with $\epsilon > 0$. In the context of peer-to-peer streaming and file sharing systems, erasure codes are used to implement *source coding*. The source performs an erasure coding of the content before distributing it in its swarm. Erasure coding schemes allow users to cope with lost data blocks caused by unreliable peers and network links. However, source coding cannot cope with heterogeneous peers. Any $k+\epsilon-1$ coded blocks are statistically independent of the original blocks. Accordingly, the download of an insufficient number of blocks leads to no useful information at all.

Finally, the (related) coding techniques *multiple description coding* (*MDC*) [5] and scalable coding [23]: Both coding schemes produce pictures and audio samples from subsets of data blocks. The size of the subset determines the quality. While MDC decodes any subset of data blocks, scalable coding enforces a pyramid-like structure: data is partitioned into a base layer and multiple enhancement layers. The base layer is mandatory to obtain an initial low quality result. The download of additional enhancement layers gradually improves the quality, but only if the layers beneath are available as well. While MDC is more flexible, scalable coding is more efficient. With a pyramid-like structure, scalable coding schemes more closely resemble current state-of-the-art coding schemes such as H.264 [30]. Simply speaking, H.264 employs two mechanisms. First, the current picture

[1] http://tools.ietf.org/html/draft-pantos-http-live-streaming-06