Modeling and Evaluation of a Metadata-based Adaptive P2P Video Streaming System

R. IQBAL, S. SHIRMOHAMMADI AND B. HARIRI
University of Ottawa, Ontario, Canada.

ABSTRACT: In this paper we present a multi-parent adaptive video streaming system (MAVSS). MAVSS is a cooperative video streaming system, based on the Peer-to-Peer (P2P) content distribution concept, to simultaneously adapt and stream video contents to heterogeneous users. In order to ensure video codec independence, we emphasize on structured metadata-based adaptation. Therefore, MPEG-21 generic Bitstream Syntax Description (gBSD) is selected to describe the parts of the video contents selected for adaptation operations. Additionally, without an optimal solution, it is hard to judge the efficiency of such systems. Therefore, in this paper, we first present the fundamental properties of a multi-parent adaptive video streaming system. We then present a mathematical model of the adaptive streaming system, where we define efficient streaming as an optimization of a cost function that can be solved as an Integer Linear Programming problem. The model illustrates the relation among all the parameters that affect the resource contribution, resource utilization, load balancing, and service fairness. We use this model to analyze the trade-offs that exist between service fairness and system efficiency.

GENERAL TERMS: Design, Experimentation, Performance.

KEYWORDS: Integer Linear Programming, Peer-to-peer streaming, Video adaptation

1. INTRODUCTION

Due to its huge success and popularity in real-world applications, the Peer-to-Peer (P2P) content distribution concept has received much research attention. However, P2P mobile multimedia sharing is not as popular yet due to the unavailability of lightweight and affordable solutions. On one side, smart handhelds with built-in Wi-Fi and rich multimedia capability are entering the consumer market on a daily basis, while on the other side network service providers are deploying deep packet inspection devices and policy products to throttle content specific traffic or to charge consumers for rich media contents. Limitations and variations of these devices also make it difficult to render and display regular videos on these devices like that of desktops or laptops. Eventually, diverse network types, bandwidth, data bucket costs, device variations, user preferences, and video codec types lead to manifold of possible adaptation requirements. The challenge here is to perform the adaptation operations in real-time and in a decentralized setting. In our scheme, we propose to utilize the idle

computing power and bandwidth of personal computers in order to adapt and stream videos to multimedia enabled devices having limited resources.

1.1. Motivation

In a P2P video streaming system, there is no prior knowledge of the participating peers. Therefore, the stream source must choose the video quality and size in advance. However, depending on the bandwidth/device capabilities of the participating peers, the selected quality/size may be too high or too low. If the rate is too high then the incoming peers may face denial of service. Yet again, if the rate is too low then the system resource in terms of upload-download capacity will not be fully exploited. In addition, even if a mobile device at full power and full bandwidth is capable of playing back high resolution videos (such as HD), such device does not always have full battery or sufficient bandwidth, due to mobility, and needs to receive the video at a quality that matches its current condition. For example, if the battery is low, the video quality can be somewhat decreased to stretch battery life and allow longer playback, or if the current available bandwidth does not match what is required for HD video, the video must be adapted to maintain real-time playback. Therefore, we propose to adapt rich media contents in real-time in order to meet the user preference, network characteristic, and device requirements. Traditional dedicated media adaptation and streaming server-based architectures usually suffer from high workload due to frequent requests and available capacity. Moreover, the deployment and operating costs may be viable only if supported commercially. A commonly used solution is to store multiple versions of the same content taking into consideration different network types and device capabilities. Apparently, this is not a scalable solution given numerous device types entering the market each day. In order to reduce complexity and to support the concept of Universal Multimedia Access, it is required to have an adaptation system that is generic and format independent so that all devices, platforms, and systems alike can access the media. It will also allow service providers to target a wide range of devices and networks, and minimize storage and maintenance requirements on the server side. This motivated us to propose a simultaneous adaptation and streaming strategy in order to perform spatiotemporal adaptation operations in intermediary nodes prior to transmitting the video to heterogeneous receivers.

For P2P adaptive video streaming, we need to utilize the peer resources in order to carry out the necessary adaptation and streaming operations. Therefore, it is necessary to consider a mechanism that encourages participating peers to contribute their available resources. Without such a mechanism, peers may intend to free ride or contribute at low rates. Another optional but important issue is the participants' perception of the service fairness in adaptive streaming systems. By service fairness, we mainly refer to a ratio between the quality of the

received content and the opportunity to participate in a streaming session. Moreover, there should be a mechanism to enable more peers to join the session at any instance of time. Such an option may reduce the video quality of the already connected peers momentarily, but eventually contribute to a more stable streaming session. Finally, in overlay multicasting and live streaming, peer information and network dynamics is not known a priori. Therefore, one has to resort to stochastic models to find the optimum results from the system at a given time. Based on the results, efficiency of the architecture can be assessed. This has been one of the motivations behind the mathematical model presented in this article. In the following subsection, we highlight the main contributions of our work in this article.

1.2. Contributions

In an attempt to address the challenges of adaptive video streaming, in this article, we consider our previously proposed compressed-domain video adaptation system [1] as a utility tool, and briefly explain how peers are scheduled to adapt, transmit and buffer video streams to match the varying network conditions using this tool. We then propose a metadata-based P2P adaptive video streaming system. Our proposed system extensively utilizes the MPEG-21 generic Bitstream Syntax Description (gBSD) [23] in order to achieve codec independence for adaptation and streaming. Our specific contributions are as follows:

- A **M**ulti-parent **A**daptive **V**ideo **S**treaming **S**ystem (**MAVSS**) aims at providing a firsthand experience of combining online video adaptation with streaming in Application Layer Multicast (ALM) overlays to serve heterogeneous devices including small handhelds. Participating peers act as the stream source, adaptation engine, and perform the streaming tasks.
- We emphasize on *service fairness* that refers to serving an equal percentage of all peers' requests when there is resource scarcity. In order to satisfy this goal, we add a global variable as a fairness constraint to our model. The main advantage of imposing this constraint is to maximize the overall service response by enabling equal service distribution to all the participating peers and ensuring fairness in that way.
- We formulate an effective taxation-based *minimum contribution* requirement. Minimum contribution requirement ensures that resource allocated to serve a peer is commensurate with that peer's contribution rate.
- We provide an analytical model of MAVSS based on the Integer Linear Programming (ILP) problem. The model is used to find the optimal solution at a given time and compare the efficiency of MAVSS. It also allows us to understand the trade-offs between service fairness, peer resource contribution, and

system resource usage. From the performance analysis, we have found that the efficiency of our heuristic approach closely follows the results found from the optimum solution.

1.3. Organization of the article

The rest of the paper is organized as follows: In section 2, we highlight some related works covering online adaptation, P2P streaming, peer contribution incentive, and analytical models for online video distribution. In Section 3, we give an overview of the compressed domain video adaptation scheme. In Section 4, we detail the fundamentals of our multi-parent adaptive video streaming system. We emphasize on the new additions to our previous adaptive video streaming design. The new additions encompass support for multiple parent approach, service fairness, minimum contribution requirement, and dynamic quality adjustment at new node joining. In Section 5, an ILP-based mathematical model to find the optimum overlay is introduced. In Section 6, we evaluate and compare the performance of the heuristic approach with the optimal solution, followed by concluding remarks in Section 7.

2. Related Work

Video adaptation is a promising horizon to enable ubiquitous access to multimedia contents. Researchers have proposed multiple-proxy based transcoding solutions, e.g. [3], to scale a video for heterogeneous environments. However, to save time and cost, it is practical to perform adaptation operations in one adaptation service point rather than forwarding data packets to different proxies. Moreover, existing adaptation solutions usually follow cascaded decoding-adaptation-reencoding steps, e.g. [4], which may not be suitable for real-time live videos. Our compressed-domain adaptation framework [1] is the very first 3-in-1 benchmark to adapt, encrypt, and authenticate video contents in any MPEG-21 compliant host where ordinary personal computers can do the required adaptation in real-time. Indeed using MPEG-21 based adaptation to support heterogeneous mobile devices has become common in recent years [46]. In this paper, our goal is to avoid dedicated adaptation servers in order to be compatible with the existing P2P streaming services such as SopCast, PPLive, PPMate, PPStream, UUSee, TVAnts and many others, and use the peers themselves for live adaptation operations. Please note that due to shortage of space, we do not discuss the basics of P2P and ALM technologies, and instead refer the readers to [35] for a comprehensive tutorial and survey on this topic.

P2P architectures have received a great deal of attention in both academia and industry, e.g. [5, 6, 7, 8, 9, 10], due to their proven scalability and alleviation of the need for powerful centralized resources. While there are many recent research works that use MPEG-21 based metadata content description for multimedia adaptation [42][47][48][49], we did not find any promising research entailing metadata-based adaptation and

overlay streaming except the PAT system, proposed by Liu et al. [11], where the authors propose a peer assisted online transcoding scheme in P2P/overlay streaming. In PAT, authors save the intermediary transcoding results as metadata and re-use it in another transcoding request. The objective behind PAT and our work are similar; however, in contrast to PAT, our system does not need metadata overlay due to the small transformation effort of video bitstream description compared to that of the video bitstream itself. In addition, without the presence of a standard metadata support, the variety of adaptation approaches in a distributed multimedia environment makes adaptation procedures complicated, not scalable (if for example different video versions are stored for different devices), and not interoperable. Finally, the presence of backup parents is not needed in our design because the tree controller re-assigns a parent when the current parent leaves the system. In PAT, authors report that metadata takes up to 6% additional bandwidth and metadata size is 20-25% of the original video file size for frame rate transcoding. In contrast, our bitstream description, in the form of MPEG-21 generic Bitstream Syntax Description (gBSD), takes up to only 7% of the compressed bitstream for spatiotemporal adaptation of the video.

In the literature, there are also prominent researches entailing live media streaming, e.g. [12]. Tan et al. [12] focus on how to construct a stable multicast tree to minimize the negative impact of frequent node departures in an overlay and how to recover from errors due to node/network failures. We handle the frequent node departure problem by applying a multiple parent approach and dividing the live stream into small clips. In their paper, Tu et al. [13] proposed an analytical framework to quantitatively study the features of a hybrid media-streaming model that involves peers, servers and directory servers. Based on this framework, an equation to describe the capacity growth of a single-file streaming system is derived and extended to multi-file systems as well as systems with peer failures. Tu et al. [13] considered streaming bandwidth available from both servers and peers. In this article, we consider both bandwidth and CPU availability in a decentralized setting eliminating the need for dedicated servers.

Impact of the characteristics of streaming source and participating peers to the perceived video quality and the playback delay has been highlighted in [14], where authors focus on the impact of the streaming rate, source upload bandwidth, and bandwidth heterogeneity on transmission and playback delay. In our performance analysis, we also show that the presence of powerful nodes with higher contribution resources in the system plays a significant role for the success of an overlay. Recent studies [5, 10] on real-world P2P streaming systems have demonstrated that the streaming performance can be typically maintained at a high level once the overlay has reached a reasonable scale in an optimum setting; however, this is challenged by peer contribution. We,

therefore, enforce a contribution rate for every peer connected to the system in order to overcome resource related issues. The idea of built-in incentives in live video dissemination is, however, not new and introduced in early researches involving live P2P video streaming, e.g. [15,16]. Studies have also shown that the tit-for-tat approach is not sufficient to prevent free-riders or fully incentivize peer, e.g. [17,18]. Tit-for-tat creates a differentiated service at the application layer, providing high-speed uploaders with short download times and low-speed uploaders with high download times. In our minimum contribution rate method, we avoid the tit-for-tat concept and consider a peer's actual capability and global knowledge of similar peers.

Finally, given a video source, a group of intended destination peers and heterogeneous network resources, the challenge is to distribute the content among these peers utilizing the available resources within the P2P paradigm. In the literature, there exist some promising solutions to address this issue; however, without an optimal solution it is hard to judge the efficiency of the existing protocols. Since the dynamic traffic of the live media stream cannot be predicted accurately, one has to resort to stochastic models, e.g. [19], for robust network optimization. Zhou et al. [20] described a stochastic model to compare different data-driven downloading strategies for P2P streaming. The model considers two performance metrics, probability of continuous playback and startup latency, to evaluate chunk selection strategies in P2P streaming. In this model, authors assume independent and homogeneous peers in a symmetric network setting. Compared to [20], the objective of this research is to model multi-parent video chunk dissemination in a P2P setting.

Our approach to online adaptation and video streaming presented in this paper is complementary to our prior works in [1, 21] where we emphasized on live adaptation in compressed domain without the need of an adaptation server. In [22], we reported an analytical model of our first adaptation and streaming system. Our previous analytical model and video distribution system were based on the single parent approach. However, a single parent based overlay suffers from two drawbacks – primarily, poor resource usage and unfair contributions due to the fact that a leaf node cannot contribute to the upload capacity. Secondly, due to the dynamic nature of nodes, the departure or failure of a top tier node can cause disruption and requires frequent refinement of the overlay. Compared to our previous work, in this paper, we present a multiple-parent based system, named MAVSS. Our work is based on ALM, and user environment characteristics are not known a priori. We also use a mathematical model to compute the optimum overlay and use the result to refine the overlay. MAVSS maximizes the overall resource usage and system efficiency by serving peer service request partially if not in whole. The new design allows multiple parents to serve a peer, and takes the resource

contribution of a peer into account to allocate the video quality to that peer in order to be fair to all the participating peers.

3. COMPRESSED-DOMAIN VIDEO ADAPTATION

Box I. Sample gBSD Representation

```
<dia:DIA
     xmlns="urn:mpeg:mpeg21:2003:01-DIA-gBSD-NS"
     xmlns:dia="urn:mpeg:mpeg21:2003:01-DIA-SpatialAdaptation-NS"
     xmlns:dia="urn:mpeg:mpeg21:2003:01-DIA-NS"
     xmlns:xs="http://www.w3.org/2001/XMLSchema"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <dia:Description xsi:type="gBSDType">
           <Header><!-- … and so on… --></Header>
           <gBSDUnit xsi:type="gbsdsa:gBSDFrameUnitType"
           syntacticalLabel=":I-Frame:0" start="576" length="34660"
           left="0" right="352" top="81" bottom="224">
           <gBSDUnit xsi:type="gbsdsa:gBSDSliceUnitType"
           syntacticalLabel=":I-Slice:0"
           start="576" length="7802" marker="disposable"
           left="0" right="352" top="0" bottom="80" />
           <!-- … and so on… -->
           </gBSDUnit>
           <!-- … and so on… -->
        </dia:Description>
</dia:DIA>
```

For compressed-domain spatiotemporal video adaptation, we rely on the MPEG-21 framework [2]. MPEG-21 Part-7 (Digital Item Adaptation, in short DIA) provides normative description formats for the media object to enable interoperability. DIA specification details the syntax and semantics of tools that may be used to assist adaptation of a Digital Item (DI). A DI is referred to as a bitstream together with all its relevant descriptions. In simplified terms, the bitstream can be audio, video, or any other media. An overview of applications making use specifically of MPEG-21 Digital Items is presented in [23]. In the specification, DIA tools are clustered into several categories. For example, Bitstream Syntax Description (BSD) describes the syntax of a binary media resource such as video. With Bitstream Syntax Description Language (BSDL), which is an XML Schema based language and standardized in the MPEG-21 framework, it is possible to design specific Bitstream Syntax Schema (BS Schema) describing the syntax of a particular coding format. A normative processor named BSDtoBin is specified in MPEG-21 to generate the adapted bitstream for BSDL. Since BSDL provides a way for describing bitstream syntax with a codec specific BS Schema, an adaptation engine consequently requires knowing the specific schema. Therefore, a generic Bitstream Syntax Schema (gBS Schema) is specified in the MPEG-21 framework to offer a format independent adaptation procedure. The gBS Schema introduces the means to describe hierarchies of syntactical units and addressing means for efficient bitstream access. A

description conforming to this schema is called a generic Bitstream Syntax Description (gBSD), which provides an abstract view on the structure of the bitstream that can be used in particular when the availability of a specific BS Schema is not ensured. The gBSD is essentially a metadata definition of the media, written in the form of XML. For example, as seen in Box I, the gBSD represents each frame's number, starting position in the bitstream, frame type, and length of each frame among other information for a video bitstream.

Box II. Sample Usage Environment Description

```
<DIA>
   <Description xsi:type="UsageEnvironmentPropertyType">
    <!-- Network Capability -->
    <UsageEnvironmentProperty xsi:type="NetworksType">
        <Network xsi:type="NetworkType">
           <NetworkCharacteristic xsi:type="NetworkConditionType"
                  maxCapacity="256000"/>
        </Network>
    </UsageEnvironmentProperty>
    <!-- Terminal Capability -->
    <UsageEnvironmentProperty xsi:type="TerminalsType">
        <Terminal id="pda">
         <TerminalCapability xsi:type="DisplayType">
         <Display>
          <DisplayCapability xsi:type="DisplayCapabilityType">
           <Mode> <Resolution horizontal="960"
                    vertical="640"/> </Mode>
          </DisplayCapability>
         </Display>
         </TerminalCapability>
        </Terminal>
    </UsageEnvironmentProperty>
   </Description>
</DIA>
```

Alongside, MPEG-21 also specifies tools to describe the usage context in the form of an Extensible Markup Language (XML) document. The Usage Environment Description (UED) covers information about the usage context, terminal capabilities (e.g. codec capabilities), network conditions (e.g. maximum or minimum bandwidth), user characteristics (e.g. personalized requirements) etc. We make use of the UED to identify device capability and network condition for adaptation requirement calculation. An example of usage environment description is given in Box II.

Compressed-domain video adaptation procedures thus can be divided into two major parts – Part-1: Compressed video and Metadata generation which is performed during the encoding phase; Part-2: Adaptation of the metadata and compressed video that is performed in an intermediary node. Tasks involved in Part-2 can be further divided into two sub-processes – a. Metadata transformation, and b. Adapted video generation based on the transformed metadata. To perform compressed-domain adaptation in an intermediary node, it is convenient to have a standard metadata description so that the video processing operations can be done in a format independent way. We emphasize on utilizing the MPEG-21 generic Bitstream Syntax Description (gBSD). gBSD is the metadata representation of the actual bitstream in the form of XML. gBSD describes

syntactical and semantic levels of the corresponding bitstream. In the gBSD, we include the hierarchical structure of the encoded video, e.g. frame numbers along with their starting byte number, length of the frame, and frame type. For each frame, we also insert the slice data information in the gBSD. The metadata for a slice includes the starting byte number of the slice, the length of the slice in bytes, and the region of the video frame where the slice belongs. We also add a marker "essential" or "disposable" for each slice.

Now, the goal of compressed-domain adaptation is to adapt video frames for a target screen size and/or framerate. To meet the user preference or device requirement, a video may be scaled down by reducing frame rate and/or spatial size. For example, an adapted version with a lower framerate can be retrieved by removing segments belonging to the temporal dimension. Similarly, a version with a lower spatial resolution is retrieved by removing segments that belong to the spatial dimension. Given the target resolution/screen size, the adaptation module will discard the disposable slices directly from the encoded bitstream. This is done by parsing the adapted metadata and then discarding those parts from the original bitstream that are missing in the adapted/transformed metadata. As discussed in [24], for spatial adaptation, it is preferable to have a smaller number of slices for each frame. The reason is that if we increase the number of slices, then it will eventually increase the overhead due to the added number of slice-headers. On the plus side, using a large number of slices in each frame gives more region-based cropping options to the adaptation module.

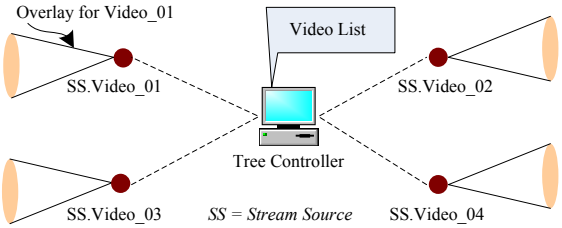
For metadata transformation, an XSLT processor is placed in the adaptation module to transform the original metadata. An XSL style sheet defines the template rules and describes how to display a resulting document. Interested readers are referred to [1] for further details on the gBSD-based video adaptation, and to [25] for UED tools and how they can assist in video streaming scenarios.

4. DESIGN OF MAVSS

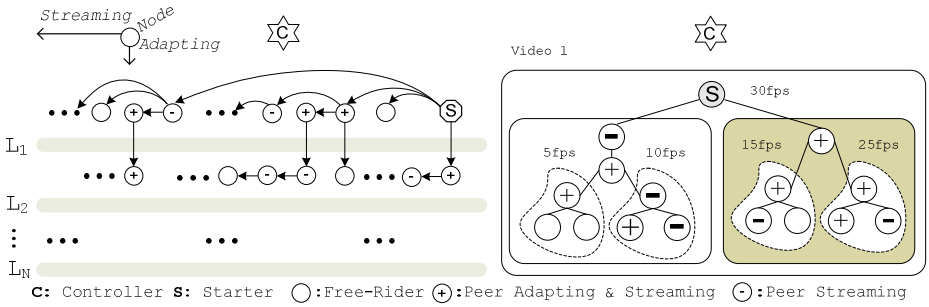
Given the global knowledge of the peers and plentiful of bandwidth and computing power, a P2P video distribution overlay design would be relatively straightforward. However, in a real scenario, designs are constrained by the limited bandwidth and computing power available with the peers. While constructing the overlay, instead of using a fixed out-degree for each node, we have computed out-degree dynamically based on the user type, content preferences, device capabilities, and the network conditions. Similar to [36] and [37], we utilize MPEG-21 to describe user characteristics, terminal capabilities, and network characteristics. Specifically, we use the UED tool to gather and express user characteristics such as content preferences and presentation preferences, descriptions of the terminal's capabilities such as display capabilities, power and storage

information, and finally network descriptions such as network end-to-end delay, bandwidth, and loss conditions. In this section, we detail different aspects of our overlay construction scheme.

4.1. Streaming Overview



A. Overlay Generation



B. Layer View C. Tree View

Fig. 1. Adaptive Video Streaming

In our adaptive streaming architecture, we follow a master-sender-driven approach. In this approach, a tree controller is responsible for administering and maintaining the video overlays whereas participating peers are responsible for video adaptation and distribution. However, the controller does not necessarily need to be a single point depending on the deployment area. There can be more than one tree controller where each tree controller is responsible for a group of peers within a region. Such systems are quite common (such as web servers, email servers, etc.) and trivial. Therefore, the system can adapt to an increase in the number of peers by adding extra tree controllers if necessary. Figure 1 shows the conceptual adaptive video streaming architecture. A separate overlay is formed for each video by the tree controller as shown in figure 1.A. Stream-starter, which can be an ordinary peer or any other media source, is the source for a specific video. Stream-starter is responsible for encoding the video and generating the corresponding metadata. Figure 1.B. is the layer view of the design, where a peer streams video to other peers on its left. A downward arrow implies the adaptation operation to serve a peer requesting lower quality video. Conceptually, the number of layers is determined by

the number of stream qualities. For example, if we consider temporal adaptation only, the system can have maximum 30 layers if the original streamed video is encoded at 30 frames per second (fps). Figure 1.C. is the tree view of the system, where we can see that a subtree is formed for each video quality.

Client-side of the application has the receiver-side and sender-side functionalities as well as an adaptation module. Small mobile devices will only have the receiving functionalities. A peer's computing (CPU) power is mapped into adaptation time to create a timing profile for each peer. Even though adaptation operations are performed based on a peer's receiving capability, a peer may request low quality video by itself. Available bandwidth and CPU power of an incoming node is collected by a daemon running on the client-side.

A new peer that has not yet joined a video overlay primarily contacts the tree controller in order to recognize its parents. Based on the incoming peer's requirements, the controller selects parents that can serve this peer possibly through the shortest (i.e. smallest) path in the tree. An incoming peer may be served by single or more parents depending on the resource availability. For the multiple parent case, each parent is responsible for delivering a certain subset of the video chunks. In the worst case, the controller attempts to assign at least one parent. Obviously, not all parents may have similar computational and networking capabilities in the multiple parent approach. Therefore, adaptation and streaming requests served will be based on the minimum the parents can support. Moreover, due to mobility, if the upload bandwidth of a parent node decreases affecting the bandwidth to serve its immediate peers, the parent will try to serve peers according to the peer class (see Table I) and joining order.

Table I. Peer Classification

Peer Class	**CPU**
Tier-1	Pentium IV, 3.0 Ghz, 1GB RAM or Higher
Tier-2	AMD Athlon XP, 1.4 Ghz, 512MB RAM or equivalent
Tier-3	Pentium III, 700 Mhz, 256 MB RAM or equivalent
Tier-4	Handheld Devices (ARM926T OMAP850 or equivalent)

It may also notify its parent(s) to reduce the frame-rate of the transmitted video. Peer information of each overlay remains sorted by the tree controller to ease the assignment of a parent in the case of sudden node departures. The tree controller also maintains an adaptation profile of all the peers. For this, a peer will adapt a small video clip for different requirements while installing the program. Based on the adaptation profile, the controller will be able to know how many adaptations this peer can perform in a certain time in Time-Division-Multiple-Access fashion. In the following subsections, we introduce the essential components of MAVSS.

4.2. Processing Video Feed

While registering for a video feed, an incoming node submits its content preference, presentation preference (e.g. color or monochrome), and processing capabilities in the UED format. Based on this information, the adaptation module computes the adaptation needs for incoming nodes. For example, if the original video is of size CIF, however, the device can support only QCIF video, then a spatial adaptation will be performed. Similarly, based on the network conditions and processing capability of the device, frame rate will be chosen and temporal adaptation may be performed.

For temporal adaptation, we drop frames from the compressed bitstream using a frame skip pattern [1], and for spatial adaptation, we drop slices outside of the region of interest (ROI) [24]. To implement our approach, we used gBSDUnitType and introduced gBSDFrameUnitType and gBSDSliceUnitType to describe each frame and slices with each frame, as shown previously in Box I. gBSDFrameUnitType consists of frame number, frame start, frame type and length of each frame for temporal adaptation. In addition to start, length and marker attributes, a gBSDSliceUnitType consists of left, right, top and bottom attributes of type nonNegativeInteger. The syntacticalLabel attribute of a frame is set to :X-Frame:Y, where X is the type of frame (i.e. I, P or B), and Y is the frame number.

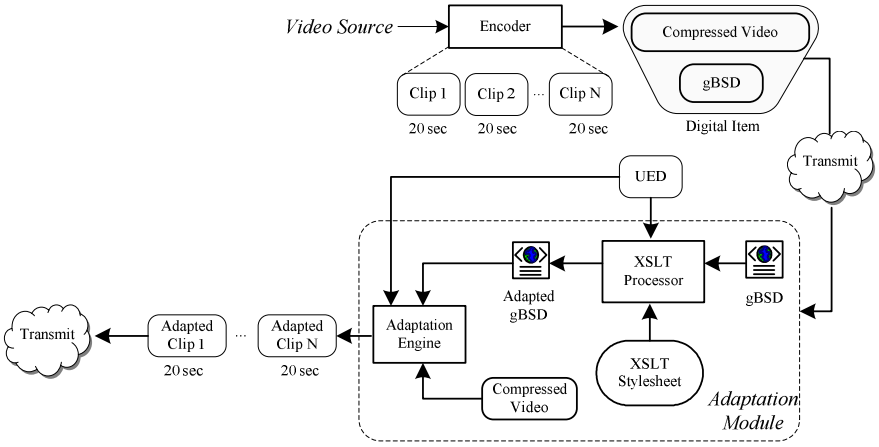


Fig. 2. Processing video feed

In Figure 2, we illustrate our compressed-domain adaptation approach. The adaptation engine performs the adaptation in two steps: first it adapts the gBSD according to the user preferences and devices characteristics extracted from the UED. This adapted gBSD is then used to generate the adapted bitstream. In spatio-temporal adaptation, adaptation characteristics for the adapted gBSD are formed in an XSL style sheet which is fed to the XSLT processor to transform the original gBSD. Template rules assembled in the style sheet are predisposed to filter the original gBSD based on the actual encoding frame rate/spatial size and target frame rate/spatial size.

To perform the actual adaptation, a peer in MAVSS can do both temporal and spatial adaptation, as described next.

Box III. Sample gBSD: Before and After Adaptation

Original gBSD for Temporal Adaptation – 5fps

```
<dia:DIA
     xmlns="urn:mpeg:mpeg21:2003:01-DIA-gBSD-NS"
     xmlns:dia="urn:mpeg:mpeg21:2003:01-DIA-SpatialAdaptation-NS"
     xmlns:dia="urn:mpeg:mpeg21:2003:01-DIA-NS"
     xmlns:xs="http://www.w3.org/2001/XMLSchema"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <dia:Description xsi:type="gBSDType">
           <Header><!-- … and so on… --></Header>
           <gBSDUnit xsi:type="gbsdsa:gBSDFrameUnitType"
           syntacticalLabel=":I-Frame:0" start="23" length="6693"/>
           <gBSDUnit xsi:type="gbsdsa:gBSDFrameUnitType"
           syntacticalLabel=":P-Frame:1" start="6716" length="6177"/>
           <gBSDUnit xsi:type="gbsdsa:gBSDFrameUnitType"
           syntacticalLabel=":P-Frame:2" start="12893" length="6186"/>
           <gBSDUnit xsi:type="gbsdsa:gBSDFrameUnitType"
           syntacticalLabel=":P-Frame:3" start="19079" length="6177"/>
           <gBSDUnit xsi:type="gbsdsa:gBSDFrameUnitType"
           syntacticalLabel=":P-Frame:4" start="25256" length="6227"/>
        </dia:Description>
</dia:DIA>
```

Adapted gBSD for Temporal Adaptation – 3 fps

```
<dia:DIA
     xmlns="urn:mpeg:mpeg21:2003:01-DIA-gBSD-NS"
     xmlns:dia="urn:mpeg:mpeg21:2003:01-DIA-SpatialAdaptation-NS"
     xmlns:dia="urn:mpeg:mpeg21:2003:01-DIA-NS"
     xmlns:xs="http://www.w3.org/2001/XMLSchema"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <dia:Description xsi:type="gBSDType">
           <Header><!-- … and so on… --></Header>
           <gBSDUnit xsi:type="gbsdsa:gBSDFrameUnitType"
           syntacticalLabel=":I-Frame:0" start="23" length="6693"/>
           <gBSDUnit xsi:type="gbsdsa:gBSDFrameUnitType"
           syntacticalLabel=":P-Frame:2" start="12893" length="6186"/>
           <gBSDUnit xsi:type="gbsdsa:gBSDFrameUnitType"
           syntacticalLabel=":P-Frame:4" start="25256" length="6227"/>
        </dia:Description>
</dia:DIA>
```

For temporal adaptation, a frame skip pattern based on the original frame rate and required frame rate (derived from the UED information) is devised to identify the appropriate nodes in the source tree of the gBSD. If the original encoding frame rate is 30 fps, the mechanism enables a consistent frame-dropping pattern for a target frame rate in between 1 to 29fps. The adapted gBSD will only consist of the frames that need to be transmitted. In Box III, we can see an example scenario where the frame rate is changed from 5 to 3 fps. In the Box, we can

see sample gBSD representations of the original and the adapted versions, where P frames 1 and 3 have been dropped in the adapted bitstream. For spatial adaptation, location and dimension of the ROI in the frame is extracted from the original gBSD and fed to the XSLT processor. The XSLT processor then generates the adapted gBSD containing the effective region for the frame based on the original spatial size and the requested spatial dimension (retrieved from UED information). As described in [24], for spatial adaptation, if any part of the slice belongs to the computed effective region, then the slice information is included in the adapted gBSD. Otherwise, the slice is omitted. In the next step, the adapted bitstream is generated using the adapted gBSD. Adapted bitstream generation consists of 3 steps - parsing the adapted gBSD, extracting the parsed gBSD information from the video and writing the adapted video stream to file. For spatio-temporal adaptation, the adaptation engine discards the frames and/or slices that are not needed in the adapted gBSD.

It should be noted that to adapt the video, an important question is what should be adapted based on the temporal, spatial, and quality modalities? For example, if a mobile client has an available bandwidth which is less than the bitrate of the original video, what type of adaption (temporal, spatial, quality, or combination) should be performed in order to match the bitrate of the adapted video to the bandwidth of the mobile client? Should we reduce the frame rate, reduce the resolution, reduce the PSNR quality, or a combination of the above? How to make this decision is a research topic of its own, and is beyond the scope of our work. We suffice it to say that many solutions for this decision making process have been proposed within the context of MPEG-21 adaptation [41][42][43] and can be applied to MAVSS as well.

Now, for the XSLT transformation of the gBSD, the complete XML description of a video file needs to be loaded before being adapted. If we generate the gBSD of a long video file then it is inconvenient to transmit the gBSD to an intermediary node for future adaptation. Therefore, we process the video feed as small clips in order to adapt the video chunks in an intermediary node with the help of gBSD information of each chunk. The video clips are encoded in the H.264 format, and alongside, their own metadata descriptions (i.e. gBSD information) are generated while encoding. Each clip and its corresponding gBSD are assigned a sequence number to represent its playback order on the receiver's side. Each video clip is individually decodable after adaptation.

For implementation convenience, we introduce 'Frameset', which refers to the number of frames equal to the frame rate. In every frameset, we ensure 1 I-frame for every 9 consecutive frames, which leads to having 3 I-frames for each second of video (and not only one I-frame). To handle multiple frame dependency, if the reference frames (I-frames or P-frames) are dropped, then the very last available I-frame of that frameset is used as the reference frame. In the case of low target frame rate (i.e. less than 10fps), the very first I-frame in that

frameset is considered as the reference frame. This avoids any perceptible distortions due to frame dropping. Now, for concurrent adaptation requests, a capable peer may need to adapt multiple framesets; hence, the total number of adaptation tasks it can perform will symbolize how many simultaneous adaptation requests it can serve per second. For continuous playback, an initial buffering enables the receivers to fill up with the received video frames for immediate and future playback, and retransmissions to another node (see figure 3). The size of the buffer depends on the average frame size, number of frames in each frameset, available memory, and delay. Now, if a node requires adapted video then its parent node(s) adapts the video before transmitting. Intermediary nodes discard the received video segments right after re-transmitting and/or adapting.

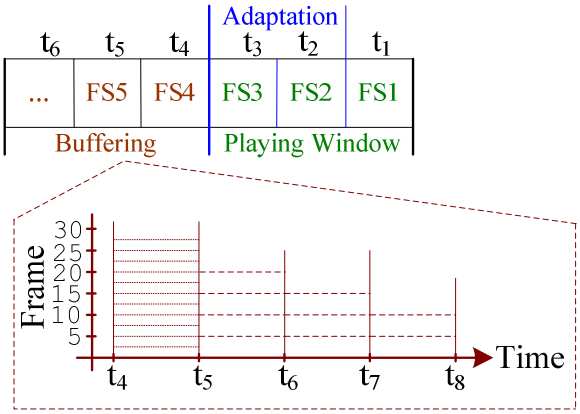


Fig. 3. Client side video Buffering, Adaptation and Playing

On the receiver side, framesets occupy the buffer, and, to this end, we use the sliding-window protocol. As shown in figure 4, a sliding window represents the active buffer portion of each peer. While viewing, a sender peer forwards the framesets from the active buffer portion to its immediate receiver peer(s) (and also adapts before transmission, if required). For a peer, if the sliding window detects too many missing framesets in the active buffer portion (up to a pre-determined limit), then either the administrative tasks (e.g. checking existence of the parent) start or the video is paused and buffering continues until the number of missing framesets drop to the acceptable limit.

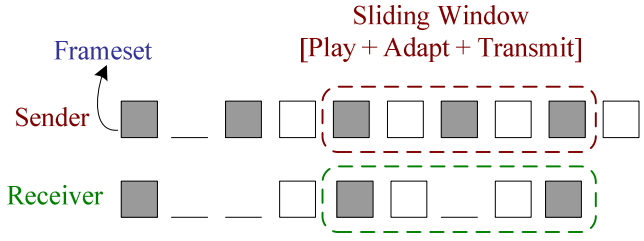


Fig. 4. Sliding window for simultaneous Adaptation and Forwarding

4.3. Node Management

As mentioned before, when a peer joins the overlay, it sends its available resource and capability information to the controller in the form of UED. Based on this information, the controller assigns this peer 'adaptation and streaming peer' role or 'streaming peer' role. An incoming peer registers itself with the tree controller and gets a list of the available streams. The tree controller attempts to connect an incoming node requesting certain video quality directly to one of the existing sub-trees (if any). If unsuccessful, a suitable node(s) will be chosen to perform the necessary adaptation in order to serve the new peer. However, if the incoming node is a free-rider, then the selected parent must be able to serve at least one more regular peer (i.e. *streaming and adaptation* or *streaming* capable peer). The tree controller stores the peer information and sends a copy of the graph to the stream source. Therefore, stream source is the secondary administrative point of contact if the tree controller fails. For further details on the node joining algorithm and node joining constraints, and parent selections, we refer interested readers to [21].

4.4. Multiple-Parent Approach

In a streaming scenario, we let a peer receive video clips from multiple higher layer peers because a single parent may not be able or willing to contribute the required outbound bandwidth or CPU in order to serve another peer completely. Such a design reduces the dependency of a receiving peer on a particular parent node, which subsequently reduces the impact due to parent departure. Additionally, it allows us to utilize the resources with fine granularity. For example, if a peer needs 150Kbps bandwidth to serve a request but it has 50Kbps available, then it will not be able to completely fulfill the request; whereas, it could offer some bandwidth to partially respond to that request. Similar scenario might also happen for CPU power. If a peer requires 2 seconds to adapt a frameset, while the operation needs to be performed in less than 1 second to meet the maximum delay constraint, then it is practical to schedule the CPU so that it can process and forward the clip after each 2 seconds. In the meantime, other peers will forward the other clips after performing the same adaptation. For obvious reasons, first layer peers are served by single parent (i.e. the stream source). However, starting from the second layer, the controller attempts to assign more number of parents for each incoming peer. If a peer is served by two parents (we call it Dual-parent approach), then odd and even framesets are sent by each of the parents as instructed by the controller, and so on. When a peer is served by more than two peers, then the controller assigns a number for each parent. Eventually, a parent forwards those framesets from a video stream that corresponds to its own number to serve a particular peer. In this regard, every peer synchronizes its virtual clock with that of the stream source. Each frameset contains originating timing information. Based on this

information, each sender peer identifies which framesets to forward. For example, let us say, a node K receives framesets from its upstream peers, and then forwards framesets to its downstream peers. For the node K being served by multiple parents ($PSet_K$), each of its parents selects and forwards framesets (FS_{send}) as follows:

$$FS_{send} = (PS_K + n \times TP_K)$$

where PS_K = serial number of the parent for receiver 'K', TP_K = total number of parents for 'K', n = frameset serial number (0,1,2,…,while true)

Peers in $PSet_K$ declare the frameset availability, and synchronize the starting frameset as follows:

$$(MaxOf(Current\_FS\_Playback(PSet_K))+1)$$

Finally, Nodes in $PSet_K$ set an offset for framesets, and start from 0 to serve K. On the receiver's side, stream startup delay occurs due to video processing operations (i.e. video capture, encode, adaptation etc), parent selection and synchronization, buffering, and transmission. To reduce this delay, for a new peer, controller initially assigns parent(s) in a sub-tree, which has a stream close to the actual requirements. Gradually, video quality improves from the next framesets.

4.5. Dynamic Quality Adjustment

In P2P streaming, start-up delay is a critical performance issue for the users who do a lot of channel surfing as well as users requesting reconnections due to peer churn and/or network dynamics. In order to tackle this issue at the joining time, we temporarily serve the joining peers with a video that is less than (or equal to, if possible) the requested quality, and readily available. Now, let Q_i denote peer i's attainable quality based on current bandwidth and processing ability, and QR_i denote the received quality of peer i at anytime. Upon joining a streaming session, peer i receives the information of its attainable quality, Q_i, from the tree controller. However, to reduce the start-up delay, tree controller assigns parent(s) to fill-up the buffer of i with the video of certain quality, QR_i, which is as close as possible to the attainable quality, i.e. $QR_i \leq Q_i$. The tree controller starts a timer for peer i, $t_i = rand(1,T)$, where T is a retry time period. The retry timer is set to be $t_i = rand(1,2^{cls}T)$, where *cls* is the peer class. Upon the expiration of the (retry) timer, the tree controller tries to assign new sources/parents to upgrade the video stream quality towards Q_i. If sufficient parents are found to serve the future video chunks with a higher quality then QR_i is updated. Meanwhile, peer i itself runs a quality decrement process to ensure that it can sustain with the expected resource contribution limit. Quality decrement process also periodically monitors the status of available resources within the peer itself and then compares the received/attainable quality with the sustainable quality at that time. If it requires updating QR_i or Q_i in order to match the resource

contribution limit then the peer informs the tree controller accordingly. In such a case, the new rate is applied to the next available frameset. Detail of the resource contribution limit is discussed in Section 4.9.

4.6. Service Fairness

In resource-scarce environments, we need to consider utilizing any available resource that can partially fulfill the demand. Even though performance should be consistent over time, we can respond to a peer join request by partially fulfilling the peer requirements in order to reduce denial of service. Moreover, we would like to degrade the service quality uniformly if there is a lack of network and CPU resources. As peers have different quality and streaming requirements depending on their viewing device, providing the same quality does not ensure fairness among the peers in terms of perceived quality. We ensure this by putting service fairness constraint, S, to all the nodes in the system for a particular overlay. For example, if a peer needs 50Kbps bandwidth from the system, but receives 25Kbps, then $S=0.5$. Therefore, S can be considered as a global fairness constraint for the entire overlay to determine the ratio between the quality of the content delivered and the quality of the content initially requested.

4.7. Node Departure

In cooperative video streaming, a peer can leave the streaming network gracefully or ungracefully by hanging, leading to service interruption when a parent leaves the overlay. Therefore, if a node is serviced by multiple parents, then the node will experience freezing of video for those framesets that were supposed to be forwarded by the departed parent. However, when the number of parents is higher (e.g. 4 or 5) then the missing frameset problem will cause comparatively less annoyance to the viewer since the frameset drop will occur after a short buffer time and will continue until a new parent has been assigned. Now, if the source node has departed the session, then that overlay is dissolved immediately. If a root node of a sub-tree has departed, then the tree controller attempts to assign a parent from the sorted availability list (from the same level or the ancestors of the disconnected parent). Recovery from sudden node departure is illustrated in figure 5 below. Now the fact cannot be avoided that sometimes there can be few good and available parent choices in the system. Therefore, if the previous parent exists in the system, then the controller tries to re-establish the connection with the previous parent.

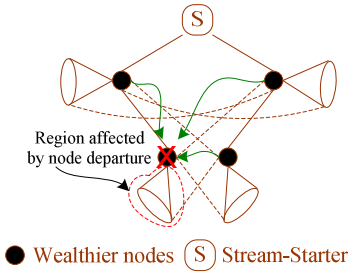


Fig. 5. Recovery from sudden node departure

4.8. Handling Packet Loss and Tree-Refinement

Consider a coded video of QCIF (176×144) resolution at a rate less than 150kbps having each P-frame in a single packet. The attempt to transmit this video over a lossy channel will obviously result in an adverse effect on the perceptual quality of the streamed video, as discussed by Liang and Apostolopoulos [26]. Therefore, in order to send the video data over the network, we encapsulate video data in packets and deliver over the IP network using UDP. Packet loss is identified through the sequence number at the receiver side. In case of high adaptation ratio, e.g. 3 frames per second (fps), if a key frame cannot be reconstructed due to packet loss during transmission, then instead of retransmitting that frame we have chosen to skip that frameset and go to the next frameset on the receiver side. In order to get optimal resource utilization and to increase reliability, we place peers with higher aptitude and stability (decided from peer history) in the upper layer of the tree. The tree refinement procedure is called when end-to-end delay of certain percentage of receiving peers exceeds the predetermined threshold for an overlay. Parent switching will take place in a way that no peer starves due to this operation. Therefore, video is prepared ahead of time by the new parent for a particular receiver peer. For example, if a peer K is to be replaced by peer C then $PSet_K$ will start serving C. Before that, a peer from the same or upper layer of $PSet_K$ will start serving K. Finally, K will be connected to C. Even though the controller manages the ALM, load imposed on the controller is relatively low as it is supposed to receive peer information only when they join the network. The controller then simply stores the information for future refinements, an operation which is linearly proportional to the number of new peers joining. Therefore, the controller's download bandwidth and/or computation power for managing information grows linearly (not exponentially) as the size of the streaming network grows.

4.9. Minimum Resource Contribution

The performance of a P2P video streaming system is highly dependent on the peer resource contribution. In [27], authors introduce a *contribution-aware* policy where the basic concept is that a peer is entitled to receive more if it contributes more. Motivated by [27], we apply a slightly different policy. We allow a peer to receive

its requested rate, and insist on it to contribute resources to the system based on its receiving rate. There is no doubt that contributions from the peers are most likely to be heterogeneous due to the heterogeneous networks in the real world. Therefore, a resourceful peer may need to contribute more compared to a resource-limited peer in order to keep the system running, and to assure that there is extra bandwidth and CPU power in the system. It should be noted that many high-speed internet users have asymmetric connections, i.e. large downloading capacity than uploading. Inevitably, an optimal overlay requires the resourceful peers to contribute more whereas those peers may choose to participate selfishly. We, therefore, try to draw a direct mapping among the actual amounts of resource consumed by a peer, peer classification and the amount of resources that the peer should contribute. Essentially, we try to facilitate equitable resource distribution among peers with similar contributions.

For ease of computing, we take a weighted sum of the bandwidth and the computing power to compute the resource availability of a peer. Resource availability (RA) of a peer i is computed from the total simultaneous adaptation and streaming requests that a peer can handle, computed as:

$$RA_i = streaming\_slot_i * 0.8 + adaptation\_slot_i * 0.2$$

We emphasize on the bandwidth because of the fact that without having enough bandwidth, we cannot forward an adapted content to another peer. Now, let RG_i be the minimum resource need to be contributed to the overlay by user i. Under a contribution rate, C, where $0 \leq C \leq 0.5$, the minimum RG be given by user i is:

$$RG_i = (L+C)\, RR_i + C * \sum(RR_i/N)$$

where L=0.5 for Tier-1, L=0.35 for Tier-2, and L=0.2 for Tier-3 peers

The minimum resource contribution rate, RG, for a peer i consists of two parts. The first term represents the minimum resource a peer is required to share by accepting resource RR. The second term is a matching rate aggregated from the minimum resource contributed by the peers from the same tier of peer i. L and C are global parameters that are known to all peers. The tree controller supplies information regarding the contribution rate of the other peers of the same tier. L is used keeping in mind the practical setting of peers from each tier. L corresponds to the following two facts – a. Same tier peers should contribute equally, and b. Less capable peer will contribute less compared to higher tier peers. We thereby ensure peer heterogeneity and fairness by forcing each peer to contribute to an extent matching the contribution of similar peers. For the contribution rate, as C approaches zero, the minimum resource contribution rate approaches to half, one third and one fifth of the resource received by Tier-1, Tier-2 and Tier-3 peer, respectively. Again, as C approaches one, the system urges the peers an impractical minimum resource contribution rate, which is nearly double of the received rate.

Therefore, we limit the value of C from 0 to 0.5. Usually, C's value will be higher when there are fewer nodes in the overlay, and it decreases as the number of peers increases and eventually their respective contribution. Finally, to utilize the contributed resources, the tree controller distributes service loads among the peers by refining the overlay so that all the peers in a same tier with similar contributed resources serve equally, which eventually ensures load balancing among serving peers.

5. ANALYTICAL MODEL and OPTIMUM OVERLAY

As we have been discussing so far, there exists a centralized node in the streaming system, denoted as tree controller, which gathers all the information from peers and decides on the overlay. At any given instant of time, a participating peer in a video overlay will receive video chunks from the source or from other peers (or both). This same peer will participate in redistributing zero or more of the received chunks to other peers. To verify the performance of this design and to calculate the optimum overlay, we followed an analytical approach based on the Integer Linear Programming method to model the MAVSS system. We use the global knowledge of the overlay collected from the tree controller to calculate the optimum value. For ease of understanding, we first model the optimum overlay maximizing the available peer resources where each receiving node is being served by a maximum one parent. Then, we revise this model to add service fairness, minimum peer contribution, and multiparent video distribution.

In a video distribution network, where the total number of peers is n for a video, cross connections for that video can be defined in a form of a matrix P, where $P:=(p_{ij})_{n\times n}$. To elaborate more, we define each of the entries in P as a decision variable to indicate if there is a connection from node i to node j, meaning that i directly serves j with no other peer nodes in-between. Value of each of these variables is a Boolean value where one (1) indicates the existence of a connection between two arbitrary nodes i and j as demonstrated in figure 6. In this matrix, the diagonal entries represent the state of connection between a node and itself. Therefore, we can remove all of these one entries from the connection matrix, which will result in a total of $(n-1)^2$ number of connection variables. The problem of optimum overlay generation can then be redefined as finding the best values for p_{ij} variables so that the overlay can take the best advantage from available resources to serve the peers.

$$\begin{pmatrix} 0 & P_{1,2} & P_{1,3} & \cdots & P_{1,n} \\ P_{2,1} & 0 & P_{2,3} & \cdots & P_{2,n} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ P_{n,1} & P_{n,2} & P_{n,3} & \cdots & 0 \end{pmatrix} \rightarrow \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ 1 & 0 & 0 & \cdots & 1 \\ \vdots & \vdots & & \cdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix}$$

Fig. 6. Connection Matrix, P

Now, the tree controller checks the availability of a specific amount of available bandwidth and processing power in the system to serve a new peer request. In order to simplify the problem, for the case of single parent approach, we assume that the incoming peers are either fully served or denied service. However, if denied, the new peer may reduce its demand and place a new service request. In Table II, we summarize the variables describing the problem parameters.

Table II. Summary of notations

ch	Refers to a chunk in the video, $ch \geq 1$
n	Total number of peers requesting chunk ch
ub_i	i^{th} peer's total upload bandwith, $i \in \{1 \ldots n\}$
db_i	i^{th} peer's required (normalized) download bandwidth for a video stream, $i \in \{1 \ldots n\}$
ap_i	i^{th} peer's available processing power, $i \in \{1 \ldots n\}$
cp_i	i^{th} peer's consuming (normalized) processing power, $i \in \{1 \ldots n\}$
In connection matrix, P, $p_{ij}=1$ iff node j is served by node i otherwise $p_{ij}=0$. $i \in \{1 \ldots n\}$, $j \in \{1 \ldots n\}$, $i \neq j$	

For a certain node, i, the amount of resource that is provided to its descendants can be defined as the sum of normalized bandwidth and processing power, R_i, over all nodes j that are serviced by i. Therefore, it can be described as below:

$$R_i = \sum_j p_{ij}(db_j + cp_j) \quad here, j=1 \ldots n, i \neq j \qquad (1)$$

Hence, the total overlay service provided by the participating nodes can be found as the sum of service provided by all single nodes as defined in equation (2)

$$R = \sum R_i = \sum_i \sum_j p_{ij}(db_j + cp_j) \quad here, i=1 \ldots n, j=1 \ldots n, i \neq j \qquad (2)$$

The problem in equation 2 can be defined as a combinational optimization problem. In an Integer Linear Programming (ILP) problem, a given linear function is maximized or minimized based on a collection of linear inequalities on a number of integer variables. A binary (0-1) Integer Linear Programming (a.k.a. Pseudo-Boolean) problem is a special case of ILP in which all variables are restricted to take values of either 0 or 1. In other words, a linear 0-1 term can either be maximized (or minimized) with respect to a set of linear constraints over 0-1 variables. An ILP problem can be written as follows:

Maximize:	The objective function, CL
	Here, C is the vector of coefficients and L is the vector of variables
Subject to:	A set of constraints, T
	Here T is defined as: $A.L \leq b$
	where, $Li \in \{0,1\}n$, and A is a matrix of coefficients
Example:	Maximize: $c_1l_1 + c_2l_2 + c_3l_3$

Subject to: $a_{11}l_1 + a_{12}l_2 + a_{13}l_3 \leq b_1$

Our goal is to find an optimum overlay in a way to maximize the total service provided in the network. Therefore, we have to find the values of *P* matrix elements in order to maximize the total service as the cost function. The problem format can be defined as an ILP problem as defined in equation 3-7 below:

Maximize

$$\sum_i \sum_j p_{ij}(db_j + cp_j) \text{ where } i=1\ldots n,\ j=1\ldots n \text{ and } i \neq j \quad (3)$$

Subject to

$$\sum_i p_{ij} \leq 1 \quad (4)$$

$$\sum_j p_{ij}\, db_j \leq ub_i \quad (5)$$

$$\sum_j p_{ij}\, cp_j \leq ap_i \quad (6)$$

$$p_{ij} \in \{0,1\} \quad (7)$$

Equation 4 above describes the fact that a requesting peer j can only be served by a single parent i. Equation 5 is describing the constraint that the total bandwidth provided by peer i cannot exceed its upload bandwidth, and equation 6 defines that the total serviced CPU power of peer i cannot exceed its total shared processing power.

Considering the above ILP model as our base model, we now extend this model for the multiple parent approach. We also consider service fairness and minimum resource contribution constraints in this revised model. In Section 4.4, we pointed out that, each video is decomposed into chunks/clips, and in the multi-parent approach, a peer receives clips of a video from multiple parents. Given that in the single parent approach, a receiving peer is solely dependent on one parent, therefore, a single connection matrix represents the overlay for the entire video. Similarly, for the multiple parent approach, we can define a connection matrix for each clip and cumulatively compute the connect matrix for the entire video.

Now, let V be the set of peers in the overlay, and the streaming source be denoted as *St*, where $St \in V$. Transmission delay $D_{st,j}$ refers to the sum of total delay from the stream source to a node j involving the intermediary nodes along the delivery path. The transmission delay should be less than a predetermined threshold, DT. In this optimization problem, as defined in Equation 8-19, we set our goal to maximize the service ratio, S, for the entire overlay where the value of S varies from 0 to 1. S determines the ratio of demand fulfillment for the peers.

Maximize

$$\text{Service Ratio, } S \quad (8)$$

Subject to

$$\sum_i P^k_{ij} \leq 1 \quad (9)$$

$$\sum_j P^k_{ij}.S.db_j \leq ub_i \quad (10)$$

$$\sum_j P^k_{ij}.cp_j \leq ap_i \quad (11)$$

$$\sum_j P^k_{ij}\,(cp_j + db_j) \geq RG_i \quad (12)$$

$\sum_j P^k_{ij} \leq |PE^k|-1$, where $(i,j \in PV^k)$, $PE^k \neq \phi$ *(13)*

$P^k_{i\,st} = 0$ *(14)*

$P^k_{ii} = 0$ if i=j *(15)*

$P^k_{ij}.D_{st,j} \leq DT$ *(16)*

$P^k_{ij} \in \{0,1\}^V$ *(17)*

$\forall k, 1 \leq k \leq ch$ *(18)*

$0 \leq S \leq 1$ *(19)*

Equation 9 above covers the facts that an incoming peer j will be served by a single parent i for a particular clip/chunk of the video. Equation 10 and equation 11, respectively, define that total bandwidth provided by peer i cannot exceed its total upload bandwidth, and that the total serviced CPU power of peer i cannot exceed its total shared processing power while serving all the children connected to it. Equation 12 imposes the constraint that each peer is forced to contribute a certain amount of resource to the network. Equation 13 ensures that there is no cycle in the overlay for each chunk. Here, PV is defined as a set of vertices, and PE as a set consisting of edges connecting elements in PV. Since, P_{ij}=1 represents a link between i and j (i.e. an edge), we can also consider P_{ij} as a decision variable. The constraint in equation 13 recursively guarantees that the number of included edges is equal to or less than the number of included vertices minus one. Equation 14 implies that there is no flow to the stream starter and Equation 15 indicates that no node transmits to itself. Equation 16 ensures that the delay threshold is maintained along the delivery path from the stream source to a receiver node.

The basic idea of solving Pseudo-Boolean constraints with 0-1 Integer Linear Programming is to solve an integer relaxation of the original problem where variables are allowed to take more values instead of being restricted only to Boolean values. A pure cutting plane algorithm can be used to reformulate the problem until the relaxation of the problem is sufficiently strong and solves the original problem [28]. This procedure therefore implies reformulating a set of constraints until 0-1 satisfiability becomes easily decidable. As an example, given a polytope and a real-valued function $f(x_1,x_2,...,x_n)$ defined on this polytope, the goal is to find a point on the polytope where this function has the largest value. In the above case, replacing the condition $L_i \in \{0,1\}^n$ by $L_i \in [0,1]^n$ is a linear programming relaxation where L can be placed on the polytope. In the literature, several methods have been used to find a solution based on the relaxation and satisfying the 0-1 constraint, e.g. Branch and bound [29], Branch and cut [30], Generalized Davis Putnam (GDP) [31], Simplex method [32], etc. However, each of these methods is useful for one specific class of problems. For example, in the GDP method, variables are iteratively chosen and removed by resolving every clause in which the variable is contained positively with any clause in which the variable is negated except this method is well suited for the problems with few variables and more constraints. In contrast, CPLEX (*www.cplex.com*) and MINOS

(*http://sbsi-sol-optimize.com/asp/sol_product_minos.htm*) solve this problem in much smaller run-time. We therefore solve the above ILP problems using the AMPL modeling language (*www.ampl.com*), and both CPLEX and MINOS solvers.

6. EVALUATION AND PERFORMANCE ANALYSIS

We have evaluated our heuristic approach aiming at finding the answer to the following questions:

- What is the real-time adaptation performance?
- What is the impact of the minimum contribution requirement?
- What is the achievable resource usage?
- What is the node joining and reconnection success rate?
- What is the impact of the dynamic quality adjustment?
- What is the impact of the service fairness on resource utilization and node joining success rate?
- What is the effect of the peer churn on the system?
- What is the overall bandwidth and CPU utilization?

We also compare the results obtained from the heuristic approach with that of the optimum overlay. Finally, we present a table showing the service fairness and its impact on the node joining success rate.

6.1. Performance Metrics

To evaluate our heuristic approach, we have defined the following performance metrics:

1. *Adaptation time profiling* - Between the two main parameters that affect the overall delay in the adaptive video streaming architecture, the network delay between peers is negligible in comparison to the adaptation time. Therefore, in this section an *adaptation time profiling* for different peer-class will be given to show the feasibility of online compressed domain adaptation.

2. *Bandwidth Service Index* and *CPU Service Index* - Bandwidth and Adaptation slot utilization provides a rough estimate of the portion of the overlay resources that has been utilized. For each peer, adaptation slot refers to the average number of adaptations a peer can perform in 1-second based on the adaptation timing profile of the peer.

Bandwidth Service Index (`BSI`) is the ratio of the number of receivers that the total overlay bandwidth could potentially sustain (at a maximum rate determined by the tree controller) to the number of actual receivers (including adaptation requests).

BSI < 1: Not all accepted peers in an overlay can receive the maximum rate

BSI = 1: The overlay is already saturated

BSI > 1: The overlay is less constrained

CPU Service Index (CSI) is the ratio of the total number of adaptation slots in the overlay to the number of adaptation slots used by the receivers.

CSI = 1: No adaptation slot available

CSI > 1: Peers requiring adaptation can join

3. *Service Disruption Index (SDI)* - Frequent node joining and leaving events (also known as *Churn*) significantly disrupts the service quality especially when the service nodes are involved. In order to evaluate the efficiency of the proposed design, a matrix named *Service Disruption Index (SDI)* has been introduced. *SDI* is the ratio of the total streaming disruption time to the whole view time since the playback begins.

4. *Node joining success rate* - The major goal of the proposed design has been the increase of the service quantity. Therefore, *node joining success rate* represents the extent of design goal satisfaction. Here, node joining includes both new incoming request and rejoin request.

6.2. Simulation Setup

We developed a video chunk level event-driven simulator in C++ to evaluate the overlay performance. We used the ANSI-C clock method to measure the timings. The trace was collected for three different video streams involving maximum 3 stream sources, 1 tree controller, and the number of concurrent peers varies from 25 to 150 for each video. The maximum number of simultaneous peers for a video has been limited to 150 for the ease of comparison between the heuristic approach and the analytical approach. Each of the nodes has been assigned the following values– node ID, peer-class, upload and download bandwidth (varies from 8Kbps to 7Mbps), and alive-time (100-time units to 5000-time units). For the video starters, we also set the corresponding video properties – video resolution (CIF 352×288, QCIF 176×144), frame rate (30fps), and average frame size (CIF: 10Kb, QCIF: 6Kb). Please note that although we have used QCIF and CIF video in our experiments, our approach is valid for any video size, for example, adapting 1080P size to 800×480. The simulator uses a random number generator to pick a node from a pool of not-connected-nodes every 5-time units. The design has been tested in both single parent scenario where only one parent serves the incoming nodes, and multiple parent scenario where a maximum of five parents serve the incoming nodes. In both cases, the participating peers are heterogeneous (in terms of available bandwidth) and demonstrate random behaviour (in terms of arrival, departure and streaming/adaptation requirements). We use UDP to transport video data, and TCP to send control packets. The results presented in the following subsection are the average results of at least 10 runs for each entry of steady state peers.

6.3. Results

Adaptation Performance of Different Peer Class - Table III presents the spatiotemporal adaptation time for each frameset utilizing MPEG-21 gBSD versus cascaded approach. The adaptation time below encompasses both metadata transformation and bit stream transformation. We have classified peers into four tiers according to their adaptation and streaming capacity. The first three tiers have exact specifications while the rest of them fall into the fourth Tier. Handheld devices are out of the scope for adaptation profiling because of their limited power and limited upload bandwidth. Such devices with limited resources are free of any streaming or adaptation operations and may join as a free-rider. From table III, we can claim that the adaptation time is quite acceptable considering the fact that processing is being done in the compressed domain and in a live fashion. Also, compared to the conventional cascaded transcoding approach, we see an order of magnitude improvement in the processing time, an amount which is consistent with the results of previous works that have used MPEG-21 based adaptation [44][45][1].

Table III. Spatiotemporal Adaptation Performance: Proposed vs. Cascaded

Peer Class	Adaptation Time (using MPEG-21)	Adaptation Time (Cascaded Approach)
Tier-1	52 - 135 Milliseconds	2.8 – 6.9 Seconds
Tier-2	128 - 300 Milliseconds	10.1 – 18.3 Seconds
Tier-3	210 - 415 Milliseconds	16.7 – 31.3 Seconds
Tier-4	*Not Applicable/Free-Rider*	

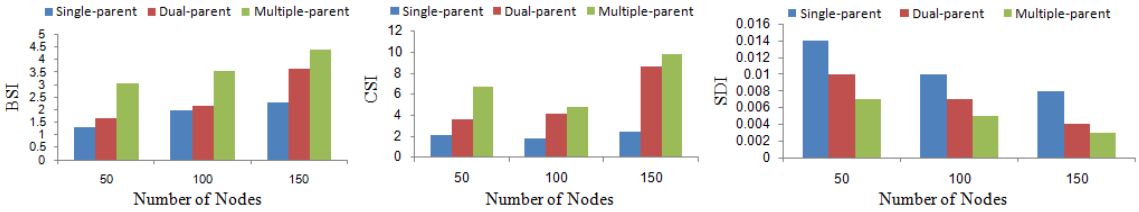


Fig. 7. BSI, CSI, and SDI for different number of parents

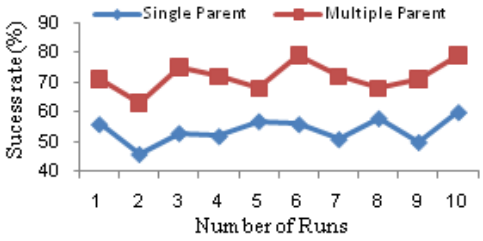


Fig. 8. Node joining success rate

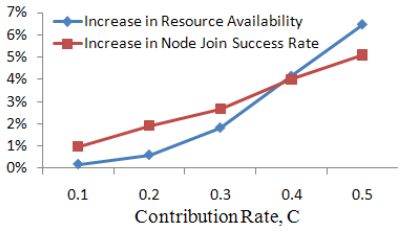


Fig. 9. Impact of the contribution rate

Performance of MAVSS - In figure 7 and figure 8, we show the resource availability and node joining success rate in the MAVSS system, respectively, without applying the service fairness and minimum contribution requirement constraints. For the results presented in figure 7 and figure 8, we consider that the presence of Tier-1, Tier-2, Tier-3 and Tier-4 peers varies from 10% to 15%, 20% to 30%, 20% to 30% and 20% to 25% respectively. From figure 7, we can see that both BSI and CSI gradually increase as the system receives more peers. As a result, multiple-parent approach can serve a larger number of peers due to virtual resource concatenation. Compared to bandwidth usage, adaptation slots utilized is comparatively low because of the fact that there can be an ordinary peer requesting low quality video, which can then serve other peers requesting the same video quality without any further adaptation need. Moreover, if a node is serving a peer with some specific spatiotemporal rate then it can serve other incoming peers requesting the same quality video, if the parent has enough upload bandwidth. For the SDI, we consider the extreme case, in which every parent of a node may depart ungracefully. A receiver peer does not starve entirely unless all of its parents have left the overlay. We see that receiving peers having multiple parents suffer less annoyance in terms of parent departure than those of having single or dual parents. From figure 8, we can observe that the node joining success rate is consistent during different runs considering the peer heterogeneity and random arrival/departure of peers. The tree refinement operation also enhances the system performance by the shifting of resourceful peers at the top of the tree. In this experiment, for the multiple parent approach, before and after refinement the highest tree depth is 7 and 5, respectively.

Effectiveness of Minimum Resource Contribution - Figure 9 shows the change in available resources and node joining success rate when a minimum level of contribution is imposed on the peers. The results demonstrate the fact that resource availability increases consistently, indicating the efficiency of the constraint to alleviate resource scarcity in the adaptive streaming system. However, node joining success rate may not increase with the same pace. This experience can be explained as follows - as the contribution rate increases, the peers become more altruistic, however, due to the peer churn or limited number of active peers, the resource sharing may not be always possible.

Effect of the Dynamic Quality Adjustment Scheme - The dynamic quality adjustment scheme benefits all the nodes that request adapted videos. Assuming a minimum of 10 framesets buffer for each node and setting the retry time, T=2 unit time, we run the simulation for ten times. Simulation results show that the quality adjustment scheme benefits on average 27% to 35% newly joined nodes where reduction in the initial wait time

ranges from 15% to 22%. On the other hand, average-waiting time to switch to the requested quality Q_i varies from 3 to 10 unit times. Longer switching times are mainly due to the peer churn and channel surfing.

Table IV. Node joining success rate for different peer classification and service ratio

Peer Classification	Service Ratio, S	Node Join Success
Tier-1: 5% Tier-2: 35% Tier-3: 35% Tier-4: 25%	0.6	83%
	0.7	81%
	0.8	76%
	0.9	73%
Tier-1: 10% Tier-2: 30% Tier-3: 35% Tier-4: 25%	0.8	84%
	0.9	81%
	1.0	76%
Tier-1: 20% Tier-2: 30% Tier-3: 30% Tier-4: 20%	0.9	88%
	1.0	84%

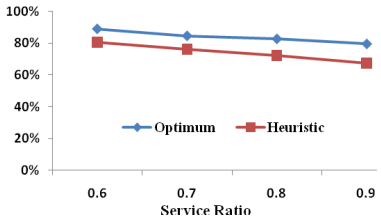


Fig. 10. Bandwidth utilization

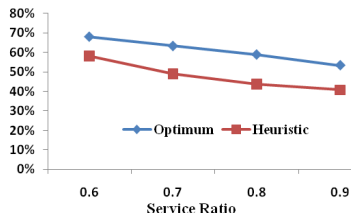


Fig. 11. CPU utilization

Effectiveness of the Service Fairness - In Table IV, figure 10 and figure 11, we report the effectiveness of the service fairness constraint as well as the effect of the wealthier nodes in the MAVSS system. From Table IV, we can see that the density of Tier-1 peers plays an important role in building a successful overlay. For example, when Tier-1 is 5%, node-join success rate we could achieve is 83% by offering 60% (S= 0.6) of the requested video quality to all peers; whereas, when Tier-1 is 20%, we could achieve a similar (84%) joining success rate by offering full rate of service (S=1.0); Additionally, the optimum overlay was able to accommodate all the nodes offering 90% of the requested video quality to all peers. In general, as we decrease S, we can accommodate more peers and hence, increase the utilization of the system resources. As can be seen from figure 10 and figure 11, for the heuristic approach, bandwidth and CPU utilizations are consistently over 67% and 40%, respectively. If we consider peer mobility and peer disconnections, which is typical in any P2P system, then the overall resource utilization indicate the efficiency of the heuristic approach. In addition, we can see that

the performance of the heuristic approach closely follows the resource utilization boundary of the optimal overlay in terms of both bandwidth and CPU utilization.

7. CONCLUSION

Until now, video sharing/streaming concept has been limited to sharing only bandwidth, and little has been done to investigate the utilization of idle CPU resources of the participating peers. While still not of the highest quality, first generation platforms like SopCast, PPMate supports live P2P video streaming sessions with a large number of global users who are using ordinary personal computers and laptops. The P2P video streaming community now needs to move towards supporting mobile and heterogeneous devices. Therefore, adding the functionality of content adaptation to ordinary video streaming applications is a timely initiative to open up the race for second-generation video streaming systems. In this paper, we have shown that if the participants in a network agree to not only share their bandwidth, but also their computing resources, then heterogeneous devices can be accommodated ensuring adequate resource utilization, with respect to resilience to peer and network dynamics. We presented design principles for a multi-parent cooperative adaptive video streaming system. In order to balance between efficient resources usages, we have added service fairness and minimum resource contribution requirement to the problem. The ILP-based mathematical model helps us to judge the efficiency of the system at any given time. Benefit of the multiple parent approach and splitting video streams into framesets is that the overall resiliency of video overlays is improved since a node will not completely starve by the failure of a parent. As losses are often temporarily correlated along each path, splitting the video framesets between different independent routes can be seen as a way to protect bitstream from consecutive losses. Although layered or scalable video coding (SVC), e.g. the work in [33], could be an option instead of video adaptation, we may not achieve a fine grained adaptation performance in layer encoded video since the more the number of layers, the less the coding efficiency. On the other hand, there are recent approaches that combine both layered and metadata-based content description, specifically SVC and MPEG 21 [38][39][40], indicating that the combination of the two paradigms is feasible. Additionally, stream starter will require high CPU load to encode the scalable video. On the other hand, benefit of H.264/AVC coding is that there is no layer dependency and all of the overlays can have equal number of peer contribution. Otherwise, overlays for base layers have to have higher number of peers than that of higher layers. The application of our adaptation system and proposed analytical model is not only limited to video streaming systems but also for a distributed camera network architecture supporting device heterogeneity, like [34]. Finally, the practical impact of the entire proposal might

be limited if the participating peers do not share their idle resources honestly or content distribution stakeholders are reluctant to adopt the MPEG-21 framework.

REFERENCES

1. Iqbal, R., Shirmohammadi, S., El Saddik, A., and Zhao. J. (2008) Compressed Domain Video Processing for Adaptation, Encryption, and Authentication. IEEE Multimedia, 15(2), 38-50.
2. ISO/IEC 21000-7:2004 (2004) Information Technology Multimedia Framework Part 7: Digital Item Adaptation.
3. Hossain, M.S., Alamri, A., and El Saddik, A. (2008) QoS-Aware Service Selection for Multimedia Transcoding. In Proceedings of IEEE I^2MTC, Victoria, BC, 12-15 May, pp. 588-593.
4. Wang, Y., Fan, X., Li, H., LIU, Z., and LI, M. (2006) An Attention Based Spatial Adaptation Scheme for H.264 Videos on Mobiles. Proceedings of Multi-Media Modeling Conference, 4-6 Jan.
5. Li, B., Xie, S., Qu, Y., Keung, Y., Lin, C., Liu, J., and Zhang, X. (2008) Inside the New Coolstreaming: Principles, Measurements and Performance Implications. Proceedings of IEEE INFOCOM, Phoenix, Arizona, USA.
6. Decuetos, P. and Ross, K.W. (2004) Unified Framework for Optimal Video Streaming. In Proceedings of IEEE INFOCOM, Hong Kong, 7-11 March, Vol. 3, pp. 1479-1489.
7. Jennigs C. and Bryan, D. (2006) P2P for Communications: Beyond File Sharing. In Business Communication Review.
8. Magharei, N., Rejaie, R. and Guo, Y. (2007) Mesh or Multiple-Tree: A Comparative Study of Live P2P Streaming Approaches. Proceedings of IEEE INFOCOM, 6-12 May, pp. 1424-1432
9. Magharei, N., Rejaie, R. (2009) PRIME: Peer-to-Peer Receiver Driven Mesh-based Streaming. IEEE/ACM Trans. Netw. 17(4), 1052-1065.
10. Hei, X., Liang, C., Liang, J., Liu, Y., and Ross, K. (2007) A Measurement Study of a Large-Scale P2P IPTV System. IEEE Trans. MM, 9(8), 1672-1687.
11. Liu, D., Setton, E., Shen, B., and Chen, S. (2007) PAT: Peer-Assisted Transcoding for Overlay Streaming to Heterogeneous Devices. Proceedings of NOSSDAV, Urbana-Champaign, IL, USA, 4-5 June.
12. Tan, G., Jarvis, S. A., and Spooner, D. P. (2007) Improving the Fault Resilience of Overlay Multicast for Media Streaming. In IEEE Trans. Parallel and Distributed Systems, 18(6), 721-734.
13. Tu, Y., Sun, J., Hefeeda, M., Xia, Y., and Prabhakar, S. 2005. An Analytical Study of Peer-to-Peer Media Streaming Systems. ACM TOMCCAP, 1(4), 354-376.
14. Ouali, A., Jaumard, B., and Hebuterne, G. (2008) Trade-offs in Peer Delay Minimization for Video Streaming in P2P Systems. Proceedings of IEEE International Symposium on Cluster Computing and the Grid (CCGRID). Lyon, France, 19-22 May, pp. 615-620, IEEE Computer Society, Washington, DC, USA.
15. Padmanabhan, V.N., Wang, H.J., Chou, P.A., and Sripanidkulchai, K. (2002) Distributing streaming media content using cooperative networking. Proceedings of International workshop on Network and operating systems support for digital audio and video (NOSSDAV), Monterey, California, USA, 1-3 June, pp. 177-186, ACM, New York, NY, USA.
16. Venkataraman, V., Francisy, P., and Calandrino, J. (2006) Chunkyspread: Heterogeneous unstructured tree-based peer-to-peer multicast. Proceedings of the IEEE International Conference on Network Protocols, Santa Barbara, California, USA, 12-15 November, pp. 2-11, IEEE Computer Society, Washington, DC, USA.
17. Piatek, M., Isdal, T., Anderson, T., Krishnamurthy, A. and Venkataramani, A. (2007) Do incentives build robustness in BitTorrent?. Proceedings of the 4th USENIX conference on Networked systems design & implementation (NSDI). Cambridge, MA, USA. 11-13 April.
18. Sirivianos, M., Park, J.H., Chen, R. and Yang, X. (2007). Free-riding in BitTorrent networks with the large view exploit. Proceedings of International Workshop on Peer-to-Peer Systems (IPTPS), Bellevue, WA, USA, 26-27 February, Microsoft Research.
19. Bertsimas, D. and Tsitsiklis, J. (1997) Introduction to Linear Optimization. Athena Scientific Press.

20. Zhou, Y., Chiu, D., and Lui, J. (2007) A Simple Model for Analyzing P2P Streaming Protocols. Proceedings of ICNP, Beijing, China, 16-19 October, pp. 226-235.
21. Iqbal, R. and Shirmohammadi, S. (2009) DAg-stream: Distributed Video Adaptation for Overlay Streaming to Heterogeneous Devices. Springer's Peer-to-Peer Networking and Applications, 2(3), 202-216.
22. Iqbal, R., Hariri, B., and Shirmohammadi, S. (2008) Modeling and Evaluation of Overlay Generation Problem for Peer-assisted Video Adaptation and Streaming. Proceedings of NOSSDAV, Braunschweig, Germany, 28-30 May, pp. 87-92, ACM, New York, NY, USA.
23. Timmerer, C. and Hellwagner, H. (2010) MPEG-21 digital items in research and practice. Proceedings of the 1st International Digital Preservation Interoperability Framework Symposium, Dresden, Germany, 21-23 April, Article 8 , 8 pages, ACM, New York, NY, USA.
24. Shahabuddin, S., Iqbal, R., Nazari, A. and Shirmohammadi, S. (2009) Compressed Domain Spatial Adaptation for H.264 Video. Proceedings of ACM Multimedia, Beijing, China, 19-24 October, pp. 797-800, ACM New York, NY, USA.
25. Nam, J., Ro, Y.M., Huh, Y., and Kim, M. (2005) Visual Content Adaptation According to User Perception Characteristics. IEEE Trans. MM, 7(3), 435-445.
26. Liang, Y.J. and Apostolopoulos, J.G. (2008) Analysis of Packet Loss for Compressed Video: Effect of Burst Losses and Correlation Between Error Frames. IEEE Trans. Circuits and Systems for Video Technology, 18(7), pp. 861-874.
27. Sung, Y., Bishop, M., and Rao, S. (2006) Enabling Contribution Awareness in an Overlay Broadcasting System. Proceedings of ACM SIGCOMM, Pisa, Italy, 11-15 September, 36(4), pp. 411-422, ACM New York, NY, USA.
28. Barth, P. (1996) Logic-Based 0-1 Constraint Programming. ISBN: 0792396634, Kluwer Academic Publishers.
29. Clausen, Y.J. (1999) Branch and Bound algorithms - Principles and Examples. Dept. of Comp. Sc., University of Copenhagen, Denmark.
30. Mitchell, J. E. (2001) Integer programming: Branch-and-cut algorithms. In The Encyclopedia of Optimization, 2, 519-525.
31. Davis, M. and Putnam, H. (1960) A Computing Procedure for Quantification Theory. JACM, 7(3), 201-215.
32. Sierksma, G. (2002) Linear and Integer Programming: Theory and Practice. ISBN: 0824706730, CRC Press.
33. Mokhtarian, K. and Hefeeda, M. (2010) Analysis of Peer-assisted Video-on-Demand Systems with Scalable Video Streams. Proceedings of the first annual ACM SIGMM conference on Multimedia systems (MMSys), Scottsdale, Arizona, USA, 22-23 February, pp. 133-144, ACM, New York, NY, USA.
34. Iqbal, R., Ratti, S., and Shirmohammadi, S. (2009) A Distributed Camera Network Architecture Supporting Video Adaptation. Proceedings of ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC), Como, Italy, August 30 – September 2, 7 Pages.
35. Hosseini, M., Ahmed, D.T., Shirmohammadi, S., and Georganas, N.D. (2007) A Survey of Application-Layer Multicast Protocols. IEEE Communications Surveys and Tutorials, 9(3), 58 – 74.
36. Angelides, M.C. and Sofokleous, A.A. (2012) A game approach to optimization of bandwidth allocation using MPEG-7 and MPEG-21. Multimedia Tools and Applications, DOI: 10.1007/s11042-011-0981-0.
37. Huang, C.M., Lin, C.W. and Yang, C.C. (2010) Mobility Management for Video Streaming on Heterogeneous Networks, IEEE Multimedia, 17(1), 24 – 32.
38. Liu, W., Liang, Y., Tan, H., and Lu, X. (2011) A video adaptation framework for SVC based on content description. Proceedings of the International Conference on Multimedia Technology, 26-28 July, pp. 120 – 124.
39. Arnaiz, L., Menendez, J.M., Jimenez, D. (2011) Efficient personalized scalable video adaptation decision-taking engine based on MPEG-21. IEEE Transactions on Consumer Electronics, (57) 2, 763 – 770.
40. Zotos, N., Xilouris, G., Shao, B., Renzi, D., Kourtis, A. (2011) Performance evaluation of H264/SVC streaming system featuring real-time in-network adaptation. Proceedings of the IEEE International Workshop on Quality of Service (IWQoS), 6-7 June, pp. 1 – 3.

41. Nur, G., Arachchi, H.K., Dogan, S., and Kondoz, A.M. (2012) Seamless video access for mobile devices by content-aware utility-based adaptation. Multimedia Tools and Applications, DOI: 10.1007/s11042-012-1120-2.
42. Xu, M., He, X., Peng, Y., Jin, J.S., Luo, S., Chia, L.T., and Hu, Y. (2012) Content on demand video adaptation based on MPEG-21 digital item adaptation. EURASIP Journal on Wireless Communications and Networking, doi:10.1186/1687-1499-2012-104.
43. López, F., Martínez, J.M., García, N. (2011) A model for preference-driven multimedia adaptation decision-making in the MPEG-21 framework. Multimedia Tools and Applications, (53)1, 181 – 211.
44. Abebe, G., Coquil, D., Kosch, H. (2011) Enhancing Semantic Video Adaptation Speed through Compressed Domain Adaptation. Proceedings of the Workshop on Multimedia on the Web (MMWeb), 8 Sept. 2011, pp. 43 – 44.
45. Eberhard, M., Celetto, L., Timmerer, C., Quacchio, E., Hellwagner, H., Rovati, F.S. (2008) An interoperable streaming framework for Scalable Video Coding based on MPEG-21. Proceedings of the International Conference on Visual Information Engineering (VIE), 29 July – 1 Aug., pp. 723 – 728.
46. Adzic, V., Kalva, H., and Furht, B. (2011) A survey of multimedia content adaptation for mobile devices. Multimedia Tools and Applications, (51)1, 379 – 396.
47. Lim, T.B., Kim, K.W., Lee, Y.J., Moon, J.W., Yoon, K. (2011) Scalable application framework to support IPTV client device independence based on MPEG-21. Proceedings of IEEE International Conference on Consumer Electronics (ICCE), 9 – 12 Jan., Las Vegas, NV, USA, pp. 859 – 860.
48. López, F., Martínez, J.M., and García, N. (2010) Towards a Fully MPEG-21 Compliant Adaptation Engine: Complementary Description Tools and Architectural Models. Lecture Notes in Computer Science, 5811(2010), 155 – 169.
49. Van Deursen, D., Van Lancker, W., De Neve, W., Paridaens, T., Mannens, E., and Van de Walle, R. (2010) NinSuna: a fully integrated platform for format-independent multimedia content adaptation and delivery using Semantic Web technologies. Multimedia Tools and Applications, 46(2-3), 371 – 398.