A System for Peer-to-Peer Video Streaming in Resource Constrained Mobile Environments

Martin Stiemerling
NEC Laboratories Europe
Kurfürsten-Anlage 36
Heidelberg, Germany
stiemerling@nw.neclab.eu

Sebastian Kiesel
NEC Laboratories Europe
Kurfürsten-Anlage 36
Heidelberg, Germany
kiesel@nw.neclab.eu

ABSTRACT

Peer-to-Peer based near-live video streaming is becoming more and more popular with users of fixed-line broadband network access, but it is mostly unavailable to mobile users, as cellular networks, such as GPRS/UMTS, cannot meet the bitrate requirements, while other wireless technologies, such as WLAN, may be fast enough but cover only very limited areas. However, there is a small but important set of scenarios, where several mobile users in close physical proximity are interested in retrieving the same content. We propose a P2P-TV system that enables them to retrieve video chunks in a cooperative way. The coordinated and efficient usage of all wireless resources available to a group of mobile hosts is the key to enable P2P-TV in mobile environments. This paper introduces our general concept. Simulation based studies are presented to assess different resource allocation strategies and to demonstrate the feasibility of our approach for delivering near-live TV in resource constrained mobile environments.

Categories and Subject Descriptors

C.2 [**Computer-Communication Networks**]: Distributed Systems

General Terms

Design, Experimentation, Performance

Keywords

Peer-to-Peer, Mobile IPTV, Cooperative Network Access

1. INTRODUCTION

The IP-based delivery of live or near-live video content is an emerging application that is quickly gaining momentum. Several distribution approaches for IPTV exist; the spectrum reaches from establishing IP-based networks that are separate from the Internet (e. g., TISPAN [4]) via classic client/server applications to P2P based video distribution in the Internet (e. g., PPLive [9]). However, these efforts currently focus mainly on users with fixed (e.g., DSL) Internet access, since live video streaming requires – depending on content type and image quality – a sustained bitrate in a range from 300 kbps to 16 Mbps, which can be delivered easily and reliably only via wireline access networks.

Considering wireless Internet access for mobile devices, users basically have the choice between high speed and good coverage area, but they cannot have both at the same time: Cellular networks such as GPRS or UMTS, cover large areas of the country (at least in many densly populated and highly developed nations), but even there may be badly covered spots and connections may be aborted, e. g., if travelling on a high-speed train through a tunnel. And more importantly, these networks usually cannot deliver bitrates suitable for video streaming to every single user. Although it is foreseeable that technological advancements such as LTE [2] will increase the maximum bitrate, the average effective bitrate per user will be unsatisfactory for video streaming due to the volatile nature of the wireless channels. In contrast, other wireless access technologies such as WiFi or WiMAX, that provide significantly higher bitrates, cover only rather small areas, e. g., at home or at train stations.

The basic idea behind our approach is very simple: use several wireless Internet access links at the same time and distribute the load among them ("channel bonding"). These links do not have to use the same technology or connect to the same network operator. Actually, diversity reduces the risk of "fate sharing", i. e., the likelihood that several links become unavailable (or very slow) at the same time. One approach would be to terminate all these connections in one mobile device with several wireless interfaces, and in fact we analyze this configuration to demonstrate the feasibility of the general concept. However, having several subscriptions to network operators at the same time would cause significant costs for a user. Therefore, we propose a scheme that allows a group of users in physical proximity to jointly access a video stream, by efficiently using the Internet access ressources each user contributes to the group.

This paper is structured as follows: Section 2 presents one use case, for illustrating the basic idea of cooperative P2P streaming and for introducing additional constraints. A systematic description of the system components in Section 2 is followed by a performance analysis in Section 3. There, simulation results demonstrate the feasibility of our approach, which is compared with related work in Section 4, before Section 5 summarizes our findings and concludes the paper.

U-NET'09, December 1, 2009, Rome, Italy.

2. COOPERATIVE P2P STREAMING

2.1 Internet access in high-speed trains

In many countries, such as Japan, France, or Germany, high-speed trains play an important role in public long-distance transit. For example, when interconnecting major German cities, an "InterCity Express 3" train runs at up to 320 km/h (200 mph) through less densely populated areas. This type of train has 460 seats, with an average utilization of 47% [12]. Therefore, finding about ten passengers on a train that are interested in the same live content (e. g., the coverage of a sports event) does not seem too unrealistic.

During the short stops at train stations and while running at rather low speeds through city centers, train passengers usually are able to access the Internet using UMTS or HSDPA. However, as soon as travelling with higher speeds the achievable data rates drop far below their optimum and often the much slower GPRS is the only available network. Figure 1 depicts the measured goodput of TCP based data transfer during a 20 minutes ride on an ICE3 train and illustrates the volatile character of the connectivity.

2.2 System Architecture

For the clarity of the description we assume a scenario as laid out above. It will be shown later that the system is more general and applicable to other use cases as well.

We assume that a group of users (nodes) is riding on the same train, and the users are interested in watching the same live video stream. The base technology for video transmission is assumed to be a mesh-based P2P streaming system with a pull algorithm for the chunk exchange between the different peers and peer discovery, such as, for instance, CoolStreaming/DONet [16].

Furthermore, we assume that every mobile device is equipped with at least two wireless interfaces, which use different technologies and have different characteristics (see Fig. 2). The first type of link offers access to the Internet and we assume that every user is willing to contribute this resource to the group, for retrieving this stream in a joint effort. These links, which we will be calling *access-link* and identify by ℓ, may be based, for example, on GPRS or UMTS. They may be constrained in the sense that (i) their nominal bitrate $\hat{B}_\ell$ may be lower than the video stream's bitrate B_v and (ii) that the actually achievable link bitrate changes over time, i.e., $0 \leq B_\ell(t) \leq \hat{B}_\ell$. The second type of link may be used to communicate with the other nodes on the same train using a technology which provides high link capacity and is free of charge (e.g., WLAN in ad-hoc mode). This type of of link will be called *sharing-link* and while its bitrate is assumed to be much higher than the video bitrate, and it does not have to provide direct Internet access.

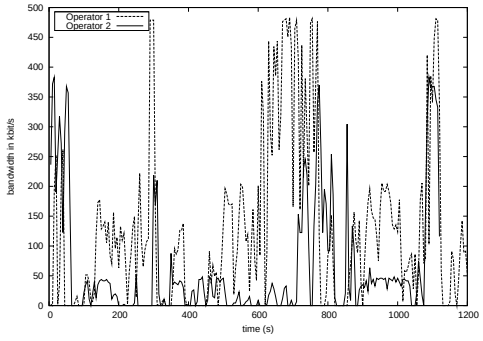


Figure 1: TCP goodput via UMTS while on a train

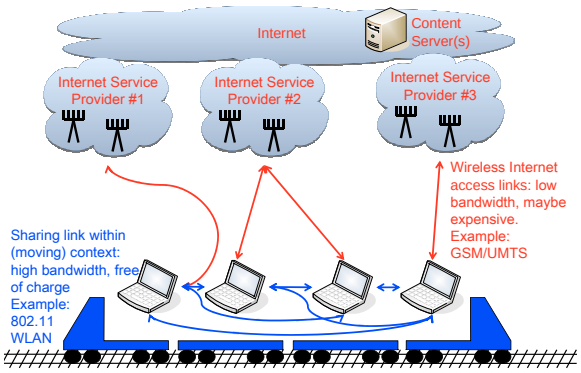


Figure 2: Usage Scenario in Train

The overall goal of the system is to efficiently leverage the combined capacity of all access-links. Each of these links will only have to download a subset of all video chunks, while the sharing-link will be used for coordination and group-internal redistribution of the chunks.

The instantaneous combined download capacity of the system is $B(t) = \sum_{\ell=0}^{L-1} B_\ell(t)$, and the system can only work stable if the average capacity is higher than the video bit rate: $\tilde{B} \geq B_\mathrm{v}$. Temporarily occurring drop outs have to be compensated by an reasonably dimensioned playout buffer, as in every P2P live streaming system. Note that the number of access-links L may be higher than the number of nodes in the group N, i.e., there may be nodes with more than one access-link. By accessing the video stream in a joint effort, the wireless access capacity of multiple nodes is effectively used.

We introduce a split-horizon for the peers, as the peers have their "view" to the Internet (via the *access-link*) with the remote peers and a "view" to their *sharing-link* where only the local peers are visible. The local peers appear as regular peers to the remote ones, with a single chunk buffer and as part of the trading logic. However, locally the peers are working on a virtual chunk buffer that is spread over all participating local peers. Each local peer maintains its own chunk buffer that contains the chunks required for the peer's operation. The sum of all peer's chunk buffers represents the current virtual chunk buffer. This split view of external to the remote and the local view is denoted as split-horizon P2P application, as each peer has two horizons.

2.3 Establishing and Maintaining the Group

Each peer has to determine if there are other peers in its vicinity to team-up with for the joint action, *before* starting to up- and download chunks via the access-link; and also to check whether the peers are still present during the system runtime (i.e., to cope with churn). For the sake of brevity we will not detail the steps necessary for the automatic configuration of the sharing-link, but instead we will be focusing on strategies for the efficient usage of the access-links.

2.4 Coordination of Chunk Retrieval

Once the participating nodes have formed a group, they need to coordinate the chunk up- and download for their links in the whole group. The overall goal is to retrieve all required chunks before the playout time of the video segment, i.e., the time when a chunk containing the part of video is required for rendering it to the user.

2.4.1 Centralized Control

This approach assumes that one peer in the group is elected as the central controller. The election could be, for example, based on peer IDs: each peer broadcasts a special "hello" message on the sharing-link, which contains the peer ID and claims that the sender is the controller. If a peer receives such a message from another peer with a lower ID contained therein, it stops broadcasting its own messages. If a peer did not receive a "hello" message with a lower ID for a reasonable period of time, it can assume that it is the controller.

The task of the central controller is to collect statistics about the achieved bitrate $B_\ell(t)$ of all L access-links in the group. Furthermore, it has to collect buffer maps from the peers outside the group, i. e., in the Internet. The controller is responsible for scheduling the chunk retrieval based on this information. That is, it asks peers in the group to retrieve specific data chunks from the indicated peer in the Internet, using the indicated access-link.

The controller is also aware of all missing and available chunks in each node's chunk buffer, which are building the virtual chunk buffer. Chunks that are present in a chunk buffer, but are missing in another chunk buffer will be exchanged via the sharing-link between the nodes. The controller makes use of the fact that the video stream uses forward error correction (FEC) [7], i.e., the controller does not necessarily need the fill the full sliding window , but only 75% of it, i.e., some chunks can be consciously skipped, if required.

2.4.2 Decentralized Control

An alternative approach is a completely decentralized coordination. In every mesh-pull based P2P streaming system, so-called buffer-map update messages play an important role. They are used for signaling to other peers which chunks are already available at a given peer and can be retrieved from there. We have a concept based on extended buffer maps, to decentralized the coordination, but this is still under evaluation.

2.5 Chunk Schedulers

In this paper, we focus on the centralized control approach and define three schedulers that are used to assign which links will handle a particular chunk. The schedulers are evaluated in Section 3.2.

2.5.1 Theoretical Scheduler

The *theoretical* scheduler assumes that the controller is able to look into the future and determine which links will be up or down and how long each phase will last. This knowledge allows the controller to obtain the best possible set of links to transfer all chunks with the cumulative bitrate. While this approach cannot be implemented in a real system, it yields the theoretical upper bound of efficiency and is therefore useful for comparing other scheduling algorithms.

2.5.2 Round-Robin Scheduler

With the *round-robin* scheduler, the controller simply picks the next available link, i.e., a link that is not busy with another chunk and the link is considered being up, for scheduling the chunk retrieval without any consideration of the past activity of the link.

2.5.3 Average Scheduler

With the *average* scheduler, the controller keeps a per link history learned from past chunk transfers. To fill the history, the controller uses first the round-robin scheduler to allow each link to build up a history. A moving average of the passive measurements of the latest chunk transmissions is considered. Out of this, the achievable bitrate and also the uptime probability availability is obtained. The links are sorted according to their uptime probability, i.e., links with a higher probability are preferred.

This scheduler explicitly considers deadline for receiving the next required chunk C_k in the chunk buffer, i.e., the time when the chunk is required to be played out, by $t_{dl}(k) = \text{ChunkRate} * (C_k - C_{played})$ with $C_k > C_{played}$. More urgent chunks are scheduled on links with a high uptime probability. A specific chunk C_k is dropped, if no link is found that is able to transmit to chunk in the required time, as the chunk would arrive too late.

The *average* scheduler considers situations where the measured bitrate is larger than the required video bitrate. The model uses the condition $\tilde{B}(t) \geq 1.2 * B_\text{v}$ to decide whether to start using *multi-load*. *Multi-load* allows the controller to assign an already scheduled chunk for a second time to a different link then used for the first time. This is described by a *multi-load* factor of 1, i.e., a single chunk may be retrieved twice. In general, a *multi-load* factor of α does allow the controller to send α chunks, each twice, out of the current trading window. *Multi-load* is beneficial for the system (see Section 3.2), but at a cost: the used bitrate towards the peers increases proportional with a factor of α, i.e., for a *multi-load* factor of α, the bitrate increase also by factor of α.

3. PERFORMANCE EVALUATION

3.1 System Model

For the purpose of the simulation we collapse the group of i nodes into a single node with a number of i access-links, i.e., a single i-times multihomed node. This simplification allows simulations towards observing effects between behavior of access-links and chunk-to-link assignments, but of course neglects effects of the sharing-link at this stage.

The system is modelled as depicted in Fig. 3 in such a way that there is a chunk generator, a single controller, L access-links, the chunk buffer, and a video player. All elements are part of the multihomed node, except for the chunk generator. The chunk generator is the source that generates chunks at a fixed rate.

The link behavior model is based on the view as seen by the P2P application, i.e., down from layer 7. The links itself are modelled by an up- and a downlink part with a peer in between. The uplink part models the behavior of the upload of chunks and sending of requests (control messages) from the local peer to remote peers. For the initial simulations, we assume that the delivery of chunks outside the local system runs smoothly, i.e., all chunks are delivered and they are

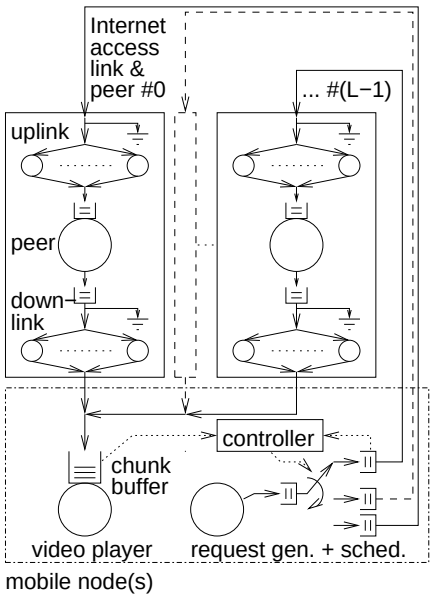


Figure 3: System Model

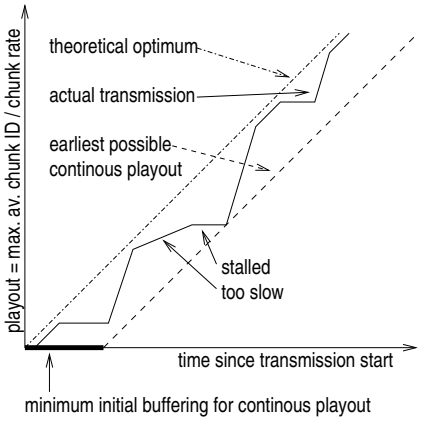


Figure 4: Minimum initial buffering time for continuous playout

delivered in-time by the peers; and chunks get only lost or are being delayed by the access-links. This step is necessary to initially evaluate the impact of our proposal and not to interleave with chunk diffusion issues outside the system.

The down link part models the behavior of the local peers and is set in such a way that there are two classes of links, (i) there are UMTS links that have a nominal link speed of 350 kbit/s and (ii) there are GPRS links with a nominal link speed of 50 kbit/s. In the first modelling the total number of links is split in 50% GPRS links and 50% UMTS links. Both classes of links, are following a negative exponential distribution for their up- and downtime, as a first approximation. GPRS links have only very short downtimes and long periods of uptime with a uptime to downtime ratio of 10. UMTS links have equally long up- and downtimes, i.e., with a ratio of 1. The links discard chunks that are in transmission if the link is down or will go down during the transmission.

The chunk buffer models the sliding window mechanics with trading-window and playout-window and is used by the player and by the controller. The player obtains the available chunks and checks whether the required chunk is available on-time, i.e., at the actual play-out time. However, there is no real video player implemented, as the simulation does not model the actual transmission of the real video.

The simulator implements the three schedulers, as described in Section 2.5, as part of the controller. The simulator is able to use the following inputs for its algorithms to compute when which chunk if request via what link or links: the filling status of the chunk buffer (i.e., position of windows), the chunks available outside the local peers, the availability of links and the current chunks in transmission.

3.2 Simulation Results

The simulator is implemented in C++ based on the IKR Simlib [11] which is an event based simulator environment. The simulations used a video bitrate B_v of 600 kbit/s (a typical P2P streaming bitrate [1]), a chunk size of 12.5 kbyte, and a total of 40,000 chunks. We have performed various simulations with changed base parameters of the access-links, a random process instead of the negative exponential distribution for the links, variation of chunk size, higher and lower video bitrate to proof that the herein presented simulation results are valid in a broader range of scenarios.

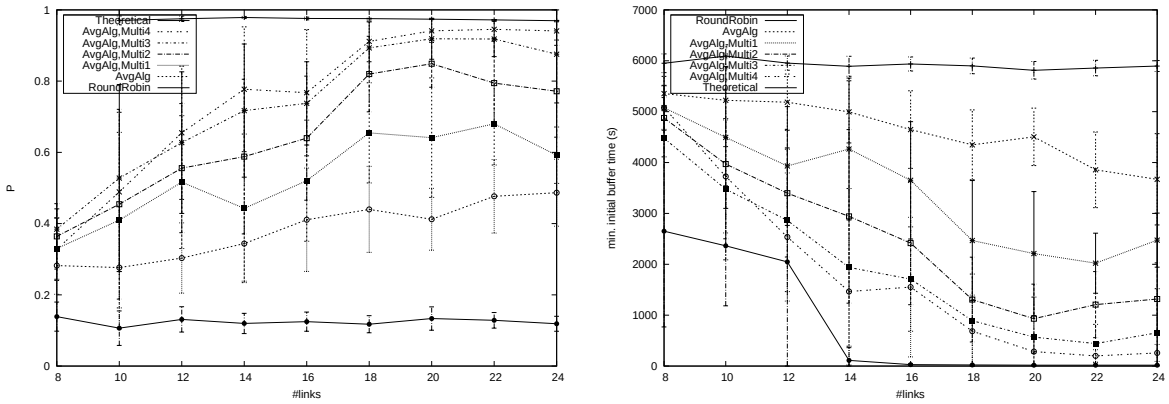
Our simulations are performed on several link sets, where the set ranges from 8 links (with normalized maximum bitrate of $\frac{\hat{B}_\ell}{B_v} = 2.67$) to 24 links ($\frac{\hat{B}_\ell}{B_v} = 8.0$); with the proposed schedulers; and with 10 different link profiles (i.e., train tracks). 8 links represents the lower bitrate bound, as the achievable bitrate over time is insufficient for too many times with less links . For all links sets, the achievable bitrate is varying in short periods in from being much higher than B_v to even lower, i.e., an insufficient amount of bitrate to receive the chunks in time at all.

For the presented link sets in Figure 5(a), we collected the information if the video player was able to retrieve a required chunk from the chunk buffer at the playout time; or that the video player stalled if there was no chunk available (see also Fig. 4). Figure 5(a) shows the chunk retrieval quality P as function of the number of links for each scheduler. P is calculated in this way, that the chunk playout at the source ($C_{playout}(t)$) is taken as reference and all results of the schedulers ($C(t)$) are normalized to the source:

$$P_{video} = \frac{\int_0^{t_{end}} C(t)dt}{\int_0^{t_{end}} C_{playout}(t)dt} \quad (1)$$

A value of P equal or close to 1 indicates that almost all chunks have been successfully transmitted and that the video can be rendered. A value of $P << 1$ indicates that most of the chunks have not been transmitted to the peers and that the video cannot be rendered.

Figure 5(a) shows that P (averaged over 10 runs) improves with an increasing amount of links, except for the *round-robin* scheduler. *Round-robin* performs bad, as this scheduler does not consider any link characteristics. The *theoretical* scheduler is always close the source playout, i.e., it can obtain almost all chunks. But for most link sets the achievable bitrate is just insufficient for some periods, so that even the *theoretical* scheduler cannot obtain all chunks. The *average* scheduler without multiload improves the chunk retrieval to $P > 0.3$. A satisfying result of $P > 0.8$ chunk



(a) normalized chunk retrieval quality

(b) minimal initial buffer time

Figure 5: Simulation Results

retrieval quality is achievable with more than 18 links and the *Average* scheduler combined with a multiload factor of 3 or higher. The main reason for the need to have a quite high number of links to achieve a 80% chunk retrieval rate is that the achievable bandwith is varying a lot over short time periods. This includes including dropping below the video bitrate B_v (see also Figure 1 as measurement reference) and a frequently changing link set in-use. 18 users being interested in the same content is not that unlikely, considering the seat utilization of German high-speed trains, as mentioned in Section 2.1.

However, the above evaluation is not sufficient to judge whether the chunks have always arrived on-time, i.e., a scenario could let the whole system stall for hundreds of seconds and then obtain the chunks very bursty, as illustrated in Fig. 4. Therefore, we define a second measure to judge the effectiveness, i.e., the *minimum initial buffering time for continuous playout*. The dash-dotted line in Fig. 4 indicates the availability of video chunks at the source vs. the time since the start of the transmission. In an ideal system without transmission delays, these chunks could be played out at the destination nodes at the same time. However, depending on the availability of the wireless transmission channels and depending on the scheduling strategy, chunks arrive with varying delays at the considered destination node (solid line). The dashed line indicates the earliest possible playout of the stream. That is, the chunk buffer at the receiving node will be empty at least once, but there will be no buffer underruns and consequently the video will not stall. The bold section on the x-axis denotes the corresponding initial buffering time. It should be noted that this time can only be determined retrospectively as it requires knowledge about the whole transmission process.

The minimum initial buffer time is shown in Figure 5(b) and only the *Average* scheduler combined with a multiload factor of 4 and an amount of 22 links or higher is working sufficiently, i.e., with a minimal initial buffer lag lower than 300 s (5mins). However, this minimum initial buffer time describes only where the video can be displayed without any interruption. But typically small interruptions of the video (i.e., a missing frame or a short missing sequence) can be tolerated by the users and can lead to much smaller initial buffer times.

4. RELATED WORK

Channel bonding and link bundling, are well-known for several link-layer technologies, such as xDSL or UMTS. However, the bundling of multiple links is handled on the link layer and does not allow bundling of multiple different link-layer technologies to aggregate bitrate. On the other hand, there is IP-based multi-homing RFC 4908 [6] [13], but with a focus on general IP multi-homing support, with enough bitrate in mind, and not considering the specific needs of P2P systems while moving. Yao presents a multi-homing agent for mobile on-board communication that aims at aggregating multiple WAN links [14] but the evaluation is solely based on wireless-LAN with a small set of only two links and low travel speed. Rodriguez et al [10] consider exploiting the diversity of multiple access links for aggregating bitrate and improving for HTTP and UDP-based streaming. However, none of these considers P2P based streaming and the impact of using application-layer channel bonding in the chunk distribution. MOVi [15] addresses a similar challenge as we do, but on a different scale, for video on-demand streaming to pedestrians, using WLAN as a very specific link-layer technology. The authors propose changes to the existing WLAN specifications and to allow a mix mode unicast streaming and peer-to-peer video exchange. Another related work is PatchPeer [3] that is designed to "allow the video-on-demand system scale beyond the bitrate capacity of the server" by combining multicast with peer-to-peer technology (similar to MOVi). Related to our work is the Drive-Thru Internet approach of Ott and Kutscher [8] where mobile clients use a session protocol to maintain connectivity even in the presence of disrupted Internet connectivity. This approach is suitable for non real-time communication, such as web browsing or email, but is not suitable for live streaming, as a constant connection is required.

The related work focuses on specific link-layer technologies [15], or requires a special mobility agent [14] [10], and considers mobility patterns of pedestrians [15] [3]. Our approach considers peers moving at the speeds of 200 km/h, e.g., high-speed trains, with more restriced and challenged GPRS and UMTS links. Furthermore, there is no need in our approach for mobility agents to be deployed on the network side, as the peers are coordinating themselves. The coordination is tailored to the chunk distribution of P2P

streaming and we are not aiming at finding a general purpose solution, as in [13], which neglects application specific needs.

Topology awareness of P2P system plays a major role in our approach, as the peers have to find neighboring peers that are in the same physical area. The recent works of the IETF ALTO working group [5] are of interest for this, but are not applicable here, as nodes do want physical closeness and do care less about topological issues.

5. CONCLUSION

This paper presents a new proposal for a P2P based application layer channel bonding, that is used for a cooperative peer-to-peer video streaming in resource constrained mobile environments, to enable mobile P2P-TV. Using this approach, a group of mobile nodes, which are located in physical vicinity can jointly access a live video stream, whose bitrate requirements exceeds the access bitrate each individual node is equipped with. An initial simulation based study demonstrates the feasibility of our approach to enable delivery of near-live TV to mobile users. Our approach is not only applicable in mobile scenarios, but also for fixed line scenarios. For instance, P2P file sharing applications used on xDSL, can utilize our approach to coordinate the download between peers in a way that not every peer is downloading the same complete file. But the file is cooperatively downloaded by all local peers, lowering the resource consumption in the edge and core network of the provider.

In the future, we will extend these studies to cover some issues, which have not been addressed in detail so far. First, the link models have to be improved and we are addressing this by performing measurements of GPRS/EDGE and UMTS availability of different network operators in German high-speed trains. Second, the implications of the sharing-link have not been considered yet, e.g., nodes might join or leave the group or the impact of the sharing link's wireless nature. Third, it has to be investigated how the proposed system interacts with algorithms in the P2P protocol that ensure fairness among the nodes – so far the simulations assume that peers in the fixed part of the network are willing to give an arbitrary amount of chunks without an adequate service in return.

Acknowledgment

Martin Stiemerling and Sebastian Kiesel are partially supported by the NAPA-WINE project, a research project supported by the European Commission under its 7th Framework Program (contract no. 214412).

6. REFERENCES

[1] E. Alessandria, M. Gallo, E. Leonardi, M. Mellia, and M. Meo. P2P-TV systems under adverse network conditions: a measurementstudy. In *Infocom 2009. The 28th Conference on Computer Communications. IEEE*. IEEE, April 2009.

[2] Erik Dahlman, Hannes Ekstrom, Anders Furuskar, Ylva Jading, Jonas Karlsson, Magnus Lundevall, and Stefan Parkvall. The 3G Long-term Evolution- Radio Interface Concepts and Performance Evolution. In *2006 IEEE 63 rd Vehicular Technology Conference*, pages 7–10, 2006.

[3] Tai T. Do, Kien A. Hua1, Ning Jiang, and Fuyu Liu. PatchPeer: A scalable video-on-demand streaming system in hybrid wireless mobile peer-to-peer networks. In *Peer-to-Peer Networking and Applications Journal*. Springer, February 2009.

[4] European Telecommunications Standards Institute. Telecoms and Internet converged Services and Protocols for Advanced Networks (TISPAN) Homepage. `http://www.etsi.org/tispan/` [last checked: 2009-08-07].

[5] IETF. IETF ALTO. `http://www.ietf.org/html.charters/alto-charter.html` [last checked: 2009-08-07].

[6] K. Nagami, S. Uda, N. Ogashiwa, H. Esaki, R. Wakikawa, and H. Ohnishi. Multi-homing for small scale fixed network Using Mobile IP and NEMO. RFC 4908, IETF, June 2007.

[7] Thinh Nguyen and Avideh Zakhor. Distributed Video Streaming with Forward Error Correction. In *Proc. of Packet Video Workshop*, September 2001.

[8] Jörg Ott and Dirk Kutscher. A Disconnection-Tolerant Transport for Drive-thru Internet Environments. In *INFOCOM 2005. 24th IEEE International Conference on Computer Communications. Proceedings*, pages 1849–1862, 2005.

[9] PPLive. PPLive Web Site. `http://www.pplive.com` [last checked: 2009-08-07].

[10] Pablo Rodriguez, Ian Pratt, Julian Chesterfield, Rajiv Chakravorty, and Suman Banjeree. MAR: A Commuter Router Infrastructure for the Mobile Internet. In *Proc. of ACM Mobisys*, pages 217–230, 2004.

[11] University Stuttgart, IKR. IKRSimLib. `http://www.ikr.uni-stuttgart.de/INDSimLib/Resources/` [last checked: 2009-08-07].

[12] wiki. ICE 3 train wiki page. `http://de.wikipedia.org/wiki/ICE_3` [last checked: 2009-08-07].

[13] W. Wong, F. Verdi, and aes M Magalh˙A next generation internet architecture for mobility and multi-homing support. In *CoNEXT '07: Proceedings of the 2007 ACM CoNEXT conference*, pages 1–2, New York, NY, USA, 2007. ACM.

[14] Jun Yao, Yi Duan, and Jianyu Pan. Implementation of a Multihoming Agent for Mobile On-board Communication. In *Proceedings of VTC*, 2006.

[15] Hayoung Yoon, JongWon Kim, Feiselia Tan, and Robert Hsieh. On-demand Video Streaming in Mobile Opportunistic Networks. In *Proceedings of PERCOM '08*, pages 80–89, Washington, DC, USA, 2008. IEEE Computer Society.

[16] Xinyan Zhang, Jiangchuan Liu, and Tak shing Peter Yum. Coolstreaming/donet: A data-driven overlay network for peer-to-peer live media streaming. In *INFOCOM 2005. 24th IEEE International Conference on Computer Communications. Proceedings*.