Sustaining Ad-Driven P2P Streaming Ecosystems *A Market-Based Approach*

Sung-Han Lin, Ranjan Pal, Bo-Chun Wang, Leana Golubchik
Department of Computer Science, University of Southern California
{sunghan, rpal, bochunwa, leana}@usc.edu

***Abstract*—Inconsistent quality of service is a significant problem in P2P-based video streaming systems. Pauses in playback are common for low capacity peers as they often upload relatively little compared to high capacity peers, and thus suffer from the 'lack of reciprocity' problem. In this work, we propose an Ad-driven Streaming P2p ECosysTem (ASPECT) that aims to eliminate the problem of playback pauses by adopting 'reduced advertisement viewing duration' as a positive incentive for peers to provide high upload rates. ASPECT rewards high capacity peers by reducing their advertisement viewing duration, when they provide more opportunities for lower capacity peers to download data. We build our research problem on a utility-theoretic market-based model, where the market stakeholders consist of a content provider, an advertisement provider, and network peers. Using concepts from game theory, we determine the system parameters to reach *market efficiency*, and study the practical implications of equilibria on the satisfaction of stakeholders' interests. From a system design perspective, one of our primary goals is to compute the equilibria advertisement viewing durations, that offer sufficient incentives for network peers to continue contributing. We evaluate ASPECT through an extensive simulation-based study. The results demonstrate that ASPECT mitigates the 'playback pause' problem for peers by at least 80% compared to existing approaches, results in appropriate advertisement viewing durations for all peers based on their contributions, and at the same time generates sufficient profit for the advertiser to be part of the market.**

***Keywords*—P2P, Game Theory, QoS, Advertisement, Market**

I. Introduction

Peer-to-Peer (P2P) based video streaming systems have been developed and deployed in order to address scalability problems that exist in client-server based streaming architectures. However, the quality of service (QoS) of P2P-based approaches is highly dependent on resources available to peers, and high and consistent QoS (where video playback proceeds without pauses) is still lacking. A peer experiences video pauses when data blocks are missing from the buffer at the time they are needed for display. For instance, in a BitTorrent-like system (which is the focus in this work), two main reasons for missing blocks are (1) a poor choice of blocks requested, on the part of the block selection algorithm and/or (2) insufficient download rates (due to not receiving data from neighbors). A number of block selection algorithms - that can make appropriate block selection choices - have been developed (see Section I-C); however, a solution for insufficient download rates is still lacking, particularly for low capacity peers. We also note that the works in [1], [2] have shown that the number of peers with high upload capacities significantly affects the performance of P2P-based streaming systems, particularly when free-riders exist in the system. Thus, our goal is to motivate high capacity peers to stay in the system and improve the overall system performance. To achieve this, we maintain their QoS and provide incentives to contribute their upload resources, to increase QoS of low capacity peers.

In this work, we focus on the following goals: (1) develop mechanisms that provide sufficient download rates for low capacity peers as well as high capacity peers so as to improve overall video streaming quality, and (2) the proposed mechanisms (in some sense) should reward peers that contribute greater resources more than those who contribute fewer resources. To accomplish the first goal, our mechanism should allows high capacity peers to obtain sufficiently high download rates so that they can experience streaming (nearly) without pauses, and then (after achieving high QoS) "*release*" whatever additional download rates they might be able to obtain to low capacity peers. To achieve the second goal, we need to identify proper incentives for peers, to enable such resource re-allocation. This is not straightforward for the following reasons.

Traditional strategies in P2P sharing systems use download rates as an incentive. For instance, a Tit-for-Tat (TFT) type strategy is often successful in P2P *file-sharing* systems, where higher download rates are used as an incentive to encourage peers to contribute their upload resources, and where lack of contribution results in longer download time. However, in *streaming* systems, these types of strategies could result in unacceptable QoS with relatively frequent video pauses [3]–[5]. To address the video pause problem, many efforts have focused on incentivizing peers to contribute to the overall increase in upload bandwidth capacity. For instance, [6]–[8] use video quality as an incentive for peers to contribute. However, given how TFT-based mechanisms currently behave, increasing overall upload capacity does not always result in low capacity peers obtaining sufficient download rates for smooth playback. Often, only high capacity peers end up with the increased download rates; consequently, video pauses are still a significant problem for low capacity peers. Another problem with using video quality as an incentive is that high capacity peers have no incentive to contribute higher upload rates, beyond the point which provides them with uninterrupted streaming. Thus, if there are no other incentives beyond video quality, high capacity peers only need to contribute sufficient upload capacity to achieve download rates needed for satisfactory video quality, resulting in degraded overall system performance.

In summary, incentivizing high capacity peers to contribute greater upload resources in return for higher download rates only works upto a point in *streaming* systems (as download rates higher than the video playback do not result in higher QoS); beyond that, other forms of incentives are needed. To this end, in this work *we investigate the proper use of advertisements (and corresponding challenges) as incentives for mitigating the QoS problem in streaming P2P systems.*

A. Advertisements as Incentives

Providing advertisements (ads) to customers is a popular business model in video streaming services. Some service providers (such as YouTube and Hulu [9]) offer free on-line

video delivery services but force customers to view fixed duration of ads (i.e., of the same duration for all users) at the beginning or in the middle of a video. For instance, Hulu inserts roughly 2 minutes worth of ads every 30 minutes, and delivers on average 82.3 ads per month to each customer [10]. Each of the ads is 15 or 30 seconds long [11]. In contrast, one could consider using *variable* length ads as an incentive to contribute resources, i.e., users that contribute more resources view shorter duration ads, and users that contribute fewer resources view longer duration ads, e.g., as proposed in [12], which reward peers with high upload rates with shorter ad duration. However, due to the commonly observed *'the rich get richer and the poor get poorer'* phenomena in P2P networks, increasing overall upload capacity does not imply increasing download rates of low (upload) capacity peers, needed to reduce video pauses. As a result, peers with high upload rates are *disproportionately* rewarded with high download rates and fewer ads. Therefore, our paper focuses on the problem of increasing the download capacity for peers that (due to their low upload capacities) do not obtain sufficient download rates to reduce video pauses and improve QoS. To mitigate the free-riding effects, our proposed mechanisms are still based on a TFT-type strategy (see Section IV-A). (In general, malicious behavior is beyond the scope of this paper.) We propose an Ad-driven Streaming P2p ECosysTem (*ASPECT*) using BitTorrent as our base system, where all peers (i) pursue satisfactory QoS, and (ii) accept an appropriate amount of ads in return.

B. A Market-Based Approach

A key challenge we address in this paper is to seek a reward function that provides sufficient incentives for peers to keep contributing. A simple approach would be to reward peers with reduced ad view durations, in a manner proportional to how much download bandwidth they "released" (e.g., using parameter tuning as discussed in Section II-B). However, such a reward function may not be sufficient (e.g., maximize individual peer utilities as discussed in Section III-B) for peers to stay in the system, and more importantly for the content provider and the advertisement (ad) provider to continue providing services (e.g., reaching market efficiency as discussed in III-C). Thus, the challenge is to design a function that jointly satisfies all peers, the content provider, and the ad provider. To achieve this goal, in Section III, we view ASPECT as a market-based model, where our market consists of a single content provider, a single ad provider, and multiple network peers. Our primary intuition behind adopting a market-based approach is its potential to arrive at incentive mechanisms that enable peers to voluntarily contribute upload capacities.

In our market setting, the content provider invests in infrastructure to provide streaming services, and signs a contract with the ad provider to show a minimal length of ads to all peers. The peers play a non-cooperative game amongst themselves, each being selfish and wanting to maximize their utility, where the utility is a monotonically increasing function on download rates they receive (benefit) and the length of ads they have to view (cost). To reduce the investment and earn more profit, the content provider would want to have more peers staying in the system and contributing their bandwidth. However, as discussed above, high capacity peers are not satisfied if their ads are not significantly fewer than the ads viewed by peers whose upload capacities are significantly smaller (due to the minimal ad duration requirement from the ad provider). Therefore, from a system design perspective, the purpose of the game is to arrive at a reward function at market equilibrium (see Section III) that leads to appropriate ad durations for peers and maximizes the sum of their utilities, and at the same time individually satisfies every peer.

Conditioned on the existence of an equilibrium point existing in the peers' game, ASPECT provides significant motivation for high capacity peers to "release" their download rates in return for viewing shorter duration ads. At the same time, low capacity peers are willing to improve their QoS without significantly increasing their ad view durations. More importantly, our result achieves market success, where the content provider is able to make its desired profit while providing sufficient incentives for peers to stay in the system (e.g., more peers staying in the system leads to greater revenues from ad viewing) and at the same time not violating the agreement made with the ad provider (i.e., ensuring that a pre-specified minimal duration of ads is viewed by all peers).

C. Related Work

Providing incentives for peers to contribute their capacities, in the hope of improving QoS, is a focus of many existing efforts with a variety of approaches [6], [12]–[15]. For instance, the work in [6] focuses on coding/MDC schemes in the context of TFT-type strategies, where peers contributing higher upload rates are rewarded with higher video quality. In [13], the authors propose a score-based incentive mechanism by converting users' contributions into scores and mapping scores into ranks, used by a peer selection mechanism for choosing which neighbors to upload to. The authors of [12] propose a system using ads as an incentive, which uses a token-based scheme for trading data between peers. Peers contributing greater upload resources can obtain tokens to reduce ad viewing time. However, none of these works guarantee to improve QoS of low capacity peers. Moreover, unlike our effort (which is decentralized and uses only information local to a peer), the work in [12] requires the use of a central server and information exchange between peers.

There are also a number of efforts focusing on QoS in BT-like systems [16]–[20]. For instance, the works in [19], [20] suggest that the block selection strategies should favor blocks that are closer to the current playback point. (The works on block selection strategies are orthogonal to ours, and we believe ASPECT can be integrated with those proposed in [19], [20].) The analysis in [16] indicates that a TFT-based strategy may not be suitable for Video-on-Demand (VoD) applications, because younger peers may not have sufficient data to share with older peers, resulting in overloading of older peers. Consequently, in [17], algorithms are proposed for load balancing requests between peers. In [18], peers increase the number of neighbors chosen during random selection when such peers already have high QoS. At a high level, this is similar to our effort, in a sense of allowing peers to not be purely selfish. However, this work does not provide incentives for peers to do so, and it runs the risk of decreasing the download rates (and hence QoS) of high capacity peers, due to lowered contributions to their high capacity neighbors. In contrast, ASPECT does not decrease the high (upload) capacity peers' ability to obtain blocks in time to avoid video pauses. Rather, we provide incentives for high (upload) capacity peers to help others, when they are able to.

Another effort [21] designs a semi-distributed algorithm to optimize fairness among peers in P2P live video systems. In its optimum case, low capacity peers have to contribute all of their capacity, but high capacity peers only have to contribute a portion of their capacity (although still higher amount than that of low capacity peers) for maintaining system performance. Thus, low capacity peers (overall) contribute less than high capacity peers while obtaining the same QoS.

In summary, ASPECT differs from the above described efforts in that it reduces video pauses by "shifting" download

rates from high capacity peers to low capacity peers while encouraging this "donation" by providing fewer ads to peers who do that. The total amount of ad viewing remains constant as the peers that take advantage of these "donations" view more ads in return for improved QoS. Moreover, ASPECT not only takes peers' well being into consideration, but also that of content and ad providers by achieving market efficiency (as detailed in Section III).

D. Main Contributions

Our research contributions can be summarized as follows.

- In contrast to many other efforts, that reduce video pause occurrences by downgrading video quality for low capacity peers, we propose a mechanism (see Section IV-B) to increase opportunities for peers to obtain sufficient download rates so as to significantly reduce video pauses. This approach nearly eliminates video pauses by increasing download rates for all peers. To determine the duration of ads for peers to view, we quantify the amount of "donation" from high capacity peers and the amount of "help" to low capacity peers (see Section IV-C). All proposed mechanisms work in a completely decentralized manner, without the need of additional support from service providers. The proposed mechanism, implemented within Ad-driven Streaming P2p ECosysTem (ASPECT), allows peers to trade their capacities and ad durations.
- We mould ASPECT into a market-based model that consists of a content provider, an ad provider, and multiple peers, and show that ASPECT is able to achieve market success. Our model facilitates study of properly designed incentives needed to encourage continued contributions from peers at market equilibrium. It also shows the existence of market equilibrium points (see Section III). A reward function is used to maximize the sum utility of all peers at equilibrium and reach market efficiency (see Section III). Thus, ASPECT provides incentives to encourage high capacity peers to continue helping low capacity peers, where low capacity peers "make up" for higher capacity peers, who view shorter duration ads (in exchange for obtaining higher download rates). Moreover, ASPECT enables the content provider to achieve its desired profit by providing sufficient incentives for all peers to stay in the system without violating the agreement with the ad provider (i.e. ensuring that a pre-specified minimal duration of ads should be viewed by all peers).

II. Overview of ASPECT

In this section, we first present an overview of ASPECT that uses 'duration of advertisements' as an incentive to keep peers contributing to the system. We then describe our trading mechanism to exchange peer download rates with ad durations. In this regard, we state our proposed peer reward function, and discuss the importance and difficulty of finding such a function in our work. An illustration of ASPECT is provided in Fig. 1.

A. A Case for Bandwidth Re-allocation

In order to increase the download rates of low capacity peers for achieving better QoS, ASPECT utilizes advertisements to provide incentives.

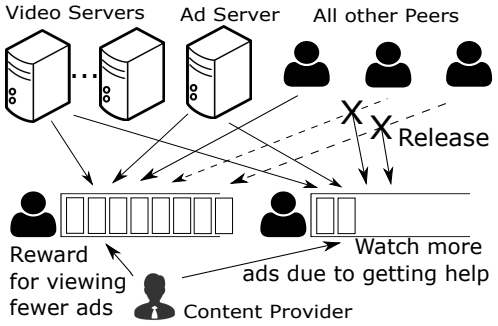

Fig. 1. Overview of ASPECT

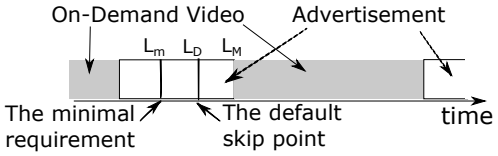

Fig. 2. On-demand Videos are Interleaved with Advertisements in ASPECT

1) Advertisement Basics: As in a regular hybrid P2P system, the content provider uses servers to deliver videos (with each video placed on one or more servers). After obtaining some initial blocks from the video servers, peers begin to exchange blocks with other peers. Here, we assume that the content provider also uses a P2P mechanism to deliver advertisements. These ads could come from an ad server, shared by many swarms (groups of peers viewing the same video). However, for ease of illustration, we only focus on one swarm in this paper. Ad blocks are shared and consumed as the video content blocks, at the same playback rate. Thus, it might also lead to experiencing video pauses during viewing ads.

Like general TV commercials, we use fixed-length ads in the middle of videos, as illustrated in Fig. 2. However, like YouTube, we allow viewers to skip the ads after a skip point (i.e., a skip button will appear). An ad provider has an agreement with the content provider for the minimal duration of ads (L_m) viewed by all peers, so the default skip point should always appear after the agreed-upon length. On the other hand, the fixed-length ads should also not exceed a common length upper bound, L_M, in order to prevent peers from leaving the system due to too much interruption. For instance, the length of current ads on TV is $\approx 31\%$ of real content [22]. Thus, the content provider has a closed interval of the ad duration at its disposal within which to operate.

2) Incentive Principle: In general, peers have heterogeneous capacities (both, upload and download rates), as illustrated in Fig. 1. In ASPECT, we provide sufficient download rates to all peers by creating the following incentive principle: *The content provider rewards peers that "release" some of the download capacity ("due to them") to the system, in exchange of shorter ad durations.* (Section IV-B describes in detail the concept of "releasing" download capacity by a peer.) *Peers that "acquire" the "released" download capacity from the system are asked to view longer ad durations by delaying the skip point.* The amount of duration the content provider advances the skip point of ads for each "acquiring" peer, is a function of the amount of download capacity "releasing" peers release to corresponding "acquiring" peers.

B. Trading Download Capacity with Advertisements

Even though we enable peers to release their download capacity in order to reduce ad lengths, they still need ap-

propriate incentives to do so. For instance, if peers cannot significantly reduce the duration of ads, they will not continue to release their download capacities. On the other hand, if peers have to view significantly longer duration ads for only a small improvement in QoS, peers might not want to stay in the system at all. *Therefore, it is important to strike an appropriate balance between ad-durations and QoS.*

1) Reward Function: For the purposes of trading download capacity with advertisements, we define a reward function to properly calculate the ad durations based on peers' contributions. We use the term 'reward function' because high capacity peers can be *rewarded* with shorter ad durations for releasing download rates ($C_n \geq 0$), and consequently, low capacity peers will be *rewarded* with better QoS by receiving download rates ($C_n \leq 0$) in return for longer ad durations. If the default ad duration before skip point is L_D, the actual skip point assigned to peer n, L_n, is calculated as:

$$L_n = \begin{cases} \min(L_D - \lambda * C_n, L_M) & \textbf{if } C_n < 0; \\ \max(L_D - \lambda * C_n, L_m) & \textbf{if } C_n > 0. \end{cases} \quad (1)$$

Here L_n is the *reward function for peer* n, and $\lambda > 0$ is the parameter used for translating download rate to ad length. For simplicity, here we use a linear function for L_n; this can be extended to a number of other functions.

2) Resolving Parameter Estimation Challenge Using Game Theory: In order to provide sufficient incentives for peers to pursue the change in ad durations, we need to find a proper combination of L_D and λ that will result in peers experiencing a sufficient QoS improvement if they view longer duration ads or have peers receive a sufficient reductions in ad durations if they release download rates. However, it is difficult to set this parameter combination. To illustrate this fact, we use a simple simulation-based experiment, based on the model discussed on Section V. This experiment uses a 30-minute video, where L_M is 7 minutes and L_m is 0. When we set $L_D = 3.5$ minutes, the actual ad durations viewed by peers with different upload capacities (under various λ values) are illustrated in Fig. 3. As we can see in Fig. 3, in some parameter combinations, peers cannot obtain too much differentiated rewards from their contributions compared to other peers' contributions. For instance, low capacity peers view all 7 minutes of ads as λ increases. This means that irrespective of the download capacity they receive, they only need to view a fixed duration of ads. High capacity peers, however, do not experience significant differences in ad durations, as a function of λ because their ad durations are already close to zero (i.e., L_m). This illustrates that an improper parameter combination leads to peers having insufficient reductions in ad durations or insufficient improvements in QoS.

Moreover, since thus far we have not considered the satisfaction of content and ad providers (which can be an important objective for sustaining an ad-driven video streaming ecosystem), an important challenge is to determine parameter values for a reward function that jointly satisfies all stakeholders, including peers, the content provider, and the ad provider. *In order to address this challenge we resort to game theory for arriving at the ideal parameter settings for the reward function, i.e., to determine appropriate L_D and λ values.*

To this end, we first model ASPECT as a market, where the goal is to design a strategic mechanism that helps us determine parameters values of the reward function, enabling all participants (the peers, the content provider, and the ad provider) to jointly satisfy their interests, given their requirements (see Section III). We then realize ASPECT in the context of a real protocol - namely a BitTorrent-like video streaming system - with modifications achieving the desired incentives (see Section IV). Our extensive simulation-based study of the resulting benefits is given in Section V.

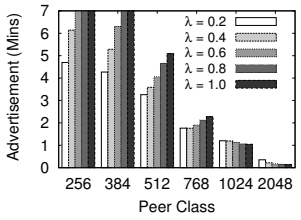

Fig. 3. Advertisements viewed as a function of λ

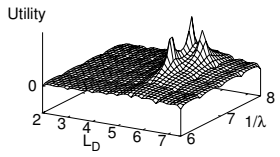

Fig. 4. Utility results from valid reward functions of advertisements

III. Market for P2P Video Streaming

Here, we design a game to determine proper parameter values for the reward function in Eq. (1) that provide sufficient incentives for peers to participate in the system and at the same time satisfies the interests of the content provider and the ad provider. We first describe the environment. Then, we formulate the peer utility functions that take individual peer download rates and ad durations corresponding to the received rates as arguments. Finally, we describe the details of the game and the notion of market efficiency.

A. The Environment

In our market setting, there is a content provider, an ad provider and a set of N peers interested in an on-demand video that streams at a constant bit rate C_v, and is provided by the content provider. The content provider invests in servers providing contents at a total upload capacity O_c. The ad provider pays the content provider for a fixed length of ads L_M, and a guaranteed minimal viewing period requirement L_m, per peer, as illustrated in Fig. 2. The content provider chooses the default skip point L_D for all peers, where $L_m < L_D$, and encourages peers to increase their contributions by enabling the skip point to appear earlier. We have a non-cooperative game among the peers, each of which is a rational, strategic player, that wants to maximize its utility (as discussed in Section III-B). Each peer acts both as a consumer, which receives video blocks from the servers and other peers, and a provider, which provides received blocks to others. Based on the received download rate D_i, peer i uses its utility function to choose the mean download rate change C_i it wants to release/receive (benefit), and the length of ads it has to view (cost).

B. Peers' Utility Functions

We define the utility of peer i as U_i, which is a function of its download rate and the length of ads it needs to view. Generally, peers have heterogeneous preferences on download rates and ads. So, we discuss them separately and combine them in the end.

As mentioned in Section I, traditional P2P mechanisms reward peers with high download rates. They believe higher rates enable peers to peruse better video qualities or to fast forward the video without freezing. However, there is typically an (upper) limit for download rates (heterogeneous for each peer) beyond which the marginal gain for peers keeps decreasing. For instance, in a constant bit rate video, peers are unlikely to consider themselves rewarded once their download rates exceed the video's bit rate. However, if the download rate is significantly lower than the video's bit rate, peers would be significantly frustrated with their low QoS, as they will experience frequent video pauses. For practical reasons, we

TABLE I. THE DISTRIBUTION OF UPLOAD BANDWIDTH

UL rate (kbps)	256	384	512	768	1024	2048
Popularity	12%	40%	31%	4%	7%	6%

model the benefit of download rates as (i) a monotonically increasing concave function (i.e., a *Cobb-Douglas* function of one variable [23], [24]), and (ii) a linear function, i.e., for peer i, U_i^{rate}:

$$\begin{cases} \textbf{(i)} \quad U_i^{rate}(b) = \begin{cases} \frac{1}{1-\gamma}(b)^{1-\gamma} & \text{if } 0 \leq \gamma < 1; \\ \log b & \text{if } \gamma = 1, \end{cases} \\ \textbf{(ii)} \quad U_i^{rate}(b) = \gamma b, \end{cases} \tag{2}$$

where γ represents a peer's preference. Here, b is the normalized received download rate:

$$b = (D_i - C_i)/f(C_v),$$

where $f(C_v)$ is the normalization function based on the video bit rate C_v. $f(C_v)$ is used to guarantee that all utilities will sit in the same range. The Cobb-Douglas and linear functions are popular in the economics literature in modeling utilities of consumers with respect to resources [24].

Peers are happier if they view shorter ad durations, and are unhappy if they have to view longer duration ads. Without loss of generality, we can also model the cost (in a sense a negative utility for a peer) of ads as a concave or linear function for peer i, $U_i^{ad}(a)$, like in Eq. (2), where a is the normalized length of an ad. Based on the amount of download capacity released/received, peers receive a deduction/increase in their ad viewing durations. Thus, a is defined as:

$$a = (L_D - \lambda C_i)/f(C_v),$$

where parameter λ "translates" download rates to ad lengths.

Based on utility functions $U_i^{rate}(\cdot)$ and $U_i^{ad}(\cdot)$, the utility function of peer i, U_i, can be defined as:

$$U_i = \alpha U_i^{rate}(b) - \beta U_i^{ad}(a) + c, \tag{3}$$

where α, β, and constant c are used to guarantee that $U_i \geq 0$.

C. Non-Cooperative Game among Peers

In this section, we first introduce the settings of the game, and then show the results of the equilibrium points reached by parameter combinations through a simulation-based study. We then discuss which combination is best suited to ASPECT.

1) Game Setting: In this game, peers can choose to stop requesting blocks from high capacity peers to "release" download rates to other peers, as discussed in Section II-A. Peers could also keep requesting blocks from high capacity peers to receive download rates released by other peers. So, in this game, the strategy parameter for each peer is its download capacity. They make this decision according to their originally received download capacities, which only depend on their upload capacities. For simplicity of the model, we assume the originally received download rate is proportional to the ratio of its upload capacity to the overall capacity. Therefore, if the upload capacity of peer i is O_i, the download rate of peer i, D_i, can be calculated as:

$$D_i = O_i/(\sum_{k=1}^{N} O_k) * (\sum_{k=1}^{N} O_k + O_c).$$

(The function for real download rates is discussed in Section IV-A.) Abstractly, there is a common-pool maintaining "free" download capacity released by peers. Peers are only allowed to release their capacities if this common-pool is empty. Peers cannot obtain download capacities beyond what this common-pool can offer. In order to achieve fairness, each peer has an opportunity to be the first to release or receive download capacities that maximize its utility in different rounds. However, due to not keeping too much free download capacity in the common-pool, peers could only change their download capacity within a fixed window in a given round. This round is repeated until the game converges to an equilibrium point.

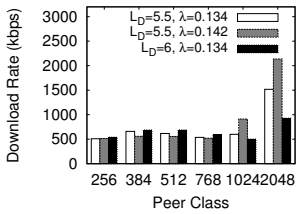

Fig. 5. The real download rates of different peer classes in three valid equilibrium points

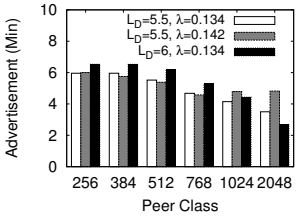

Fig. 6. The durations of ads viewed by different peer classes in three valid points

In the game, we set the number of peers to 500, all of which watch the same 30-minute, 500 kbps on-demand video. The minimal total duration of ads L_m is 1.5 minutes, and the maximal duration should not exceed 7 minutes. Peers have different upload capacities, which are drawn from the distribution given in Table I, that is based on measurements from [25]. (The same settings are used in our system experiments, discussed in Section V.) Moreover, peers have different utility functions, which are uniformly selected from all possible values of γ in U^{rate} and U^{ad}.

2) Reaching Equilibrium: An equilibrium point of our proposed game represents the point at which no peer tries to change their download rates and their ad lengths.

We iterate combinations of parameters (L_D and λ) according to the game setting in Section III-C1 and find the equilibrium point for each combination. *We observe that for each combination there is a unique equilibrium point.*

3) Reaching Market Efficiency: Market efficiency refers to the situation when all players in the market are satisfied after an economic transaction, and the market is at equilibrium. In our problem setting, the ad provider is satisfied when the length of ads shown to all peers meets the minimal requirements, in accordance with the contract with the content provider. The content provider is satisfied when peers are willing to contribute their "extra" download capacities in return for significantly differentiated ad durations. Given this, the content provider should be able to make its desired profit (e.g., more peers staying in the system lead to greater revenues from ad viewing) as well as decrease its investments in infrastructure due to significant participation of high capacity peers.

Consequently, we observe through simulation experiments that only three (L_D, λ) combinations (($L_D = 5.5$, $\lambda = 0.134$), ($L_D = 5.5$, $\lambda = 0.142$), and ($L_D = 6$, $\lambda = 0.134$)) result in market efficiency (see Fig. 4). However, out of these three efficient points, we are interested in the point that makes the content provider most satisfied. We observe this point to be ($L_D = 6$, $\lambda = 0.134$) - it has the most similar (among peers) download rates received (see Fig. 5), which should lead to similar (good) QoS as well as the most differentiated ad durations (see Fig. 6). The reason for focusing on this point in our system is that our game contains a *monopoly* content provider, and thus we choose the point where it is most satisfied. Extensions to other markets, e.g., an oligopoly, are possible but are beyond the scope of this paper.

TABLE II. SUMMARY OF NOTATIONS

$D^t_{i,n}$	the download rate from i to n in time $[t-1, t)$
$A^t_{i,n}$	the average download rate from i to n up to but not including time t
R^t_n	# of neighbors released by peer n at time t
Q^t_n	# of continuous blocks in peer n's buffer at the beginning of time t
T	length of requesting period/time slot
B	# of blocks for 1 sec of video playback
Z	the size of one block (Kb)
W^t_n	the reward for peer n at time t
P^t_n	the penalty for peer n at time t
I	the interval between two ad periods
L_D	the default duration of ads viewed by all peers in every I period
L^j_n	the actual ad duration in the j-th period
α	the coefficient used to distinguish peers groups
λ	the ratio for translating rate to ads

IV. BITTORRENT SYSTEM IN ASPECT

We gave an overview of ASPECT in Section II and studied a market-based model to find the parameters for the reward function in Section III. Here, we focus on the mechanisms that provide peers in ASPECT the ability to trade download rates and ad durations in the context of a BitTorrent-based streaming system. The BitTorrent protocol is the most popular P2P protocol and provides the general foundation of a number of widely used P2P systems today; e.g., CoolStreaming [26], the widely used block-driven P2P streaming protocol, is a BitTorrent-like protocol. (Another widely used P2P system is PPLive [27]; however, the details of its design are not publicly available.) Thus, given publically available information, we focus our design on a BitTorrent-like system.

As discussed in Section II, peers should be able to "release" unneeded download rates to low capacity peers in exchange for shorter ad durations. Thus, in this section, we first describe the components of a BitTorrent-like system that are particularly responsible for peers' download rates. Than, we introduce our proposed mechanism, with the goal of sufficiently increasing download rates for low capacity peers, but without sacrifice QoS of high capacity peers. Moreover, we propose a *decentralized* method to quantify peers' download rate changes, and combine the parameters studied in Section III to achieve differentiated ad durations.

A. BitTorrent-like Video Streaming Systems

In a *hybrid* BitTorrent-like system, peers obtain video blocks from the content provider's servers as well as exchange blocks with other peers, i.e., their neighbors in an overlay network. Peers have a strategy, to which we refer as a "Peer Selection" mechanism, for selecting from which neighbors to unchoke[1]. In order to encourage peers to contribute resources, BitTorrent-like systems typically adopt a TFT-type strategy, i.e., bigger contributors are rewarded with larger amounts of resources. For instance, a peer unchokes several (typically 4 or 5) neighbors that provide the highest download rates to it, to send blocks to. Additional peers (typically 1) are also unchoked in a random manner, in order to explore newly arrived neighbors; this is referred to as optimistic unchoking. Once a neighbor is unchoked, it needs a block selection algorithm to determine which specific blocks to request. Because the traditional rarest-first mechanism is not suitable for video streaming, other block selection algorithms [19], [20], [28], better suited for streaming, have been designed; these take the order in which blocks are streamed into consideration and thus put higher priority on selection of blocks that are needed in the near future.

Before selecting to which peer to unchoke, peers have a strategy - we refer to it as a "*Peer Request*" mechanism in the remainder of the paper - for selecting from which peers to request blocks.

Specifically, we modify the Peer Request mechanism, which is particularly responsible for download rates, and use a block selection algorithm based on the principles described in [19], as detailed in Section V.

[1]A peer is "unchoked" when it is selected to receive data (in response to its request).

TABLE III. THE AVERAGE DOWNLOAD RATES (KBPS) EXPERIENCED BY PEERS IN DIFFERENT CLASSES

Upload rate (kbps)	256	384	512	768	1024	2048
Original Download rate (kbps)	440	452	485	659	803	852
New Download rate (kbps)	525	526	528	529	546	551

B. Modified Peer Request Mechanism

Before proposing our mechanism modifications, we describe our abstraction of the BitTorrent-like video streaming system used in this paper. (A summary of notation is given in Table II.) We view the system as operating in slotted time (with a time slot of length T). At the beginning of each time slot, peers determine to whom they should send requests and which of their neighbors' requests to grant. Specifically, during time slot t, peer n has download rates, $D^t_{1,n}...D^t_{m_n,n}$[2], from its m_n neighbors. Given that a constant bit rate video requires B blocks for one time slot of playback, peer n's total download rate should be greater than or equal to the video bit rate, i.e.,

$$\sum_{i=1}^{m_n} D^t_{i,n} * T \geq B * Z, \tag{4}$$

where Z is the size of one block; otherwise, peer n might suffer from video pauses.

In order to shift un-needed download rates to low capacity peers, we take the following approach. When excess download rates (i.e. more than the required rate) are perceived by a high capacity peer n, given appropriate incentives (see Section IV-C), n forgoes on requesting blocks from *some* of its high capacity neighbors (i.e., those that provide n with high download rates). We refer to this as "releasing" a neighbor, and attempt to release as many neighbors as possible without affecting the quality of n's video playback. Specifically, we do this in an adaptive manner, where peer n increases the number of neighbors released only if its download rate satisfies the video playback requirement (i.e., no video pauses). Moreover, in order not to experience video pauses caused by insufficient download rates due to over-releasing of neighbors, peer n releases an additional neighbor only if it has already cached more than sufficient blocks in its buffer. Let Q^t_n be the number of continuous blocks[3] in peer n's buffer at the beginning of time slot t. Then, we set sufficient blocks in the buffer as $Q^t_n \geq 2*B*T$, based on the results from Experiment V-B (see details below). However, when peer n finds that the current number of released neighbors leads to insufficient download rates, i.e., Eq. (4) is not true, and the number of continuous blocks in the buffer is only sufficient to insure one (next) period for smooth playback, i.e., $2 * B * T > Q^t_n \geq B * T$, peer n decreases the number of released neighbors. Lastly, all releases are voided when peer n determines that it does not even have sufficient blocks for a single playback period, i.e., $Q^t_n < B*T$.

After determining the number of peers released, R^t_n, our mechanism sorts peer n's neighbors based on their average

[2]Given the asymmetric nature of upload/download capacities in users' connectivity, we assume (as is typically done) that the available download capacity of peers is not the bottleneck, i.e., the download rates acquired by peers are determined by the available upload capacity and used mechanisms.

[3]A set of blocks, starting from the block for the next video frame, has a continuous sequence without any missed blocks for video playback.

ALGORITHM 1: Modified Peer Request Mechanism at time t

Peer n has m_n neighbors, which are sorted on average download rates $A^t_{i,n}$ from high to low, the order is $1...m_n$;
The number of released neighbors at time $t-1$ is $R^{t-1}_n \geq 0$;
if $\sum_{i=1}^{m_n} D^{t-1}_{i,n} * T \geq BZ$ *and* $Q^t_n \geq 2BT$ **then**
  $R^t_n \leftarrow R^{t-1}_n + 1$;
else
  if $\sum_{i=1}^{m_n} D^{t-1}_{i,n} * T < BZ$ *and* $2BT > Q^t_n \geq BT$ **then**
    $R^t_n \leftarrow R^{t-1}_n - 1$;
  else
    if $Q^t_n < BT$ **then**
      $R^t_n \leftarrow 0$;
    end
  end
end
Chose requested peers from left neighbors $R^t_n + 1...m_n$;

download rates, $A^t_{i,n}$, to n and releases R^t_n peers with the highest download rates. We define $A^t_{i,n}$ as the average download rate (i.e., historical contribution), obtained by n from neighbor i (up to but not including slot t); here, we only consider download rates greater than zero when calculating this average, as we would like to reflect the average upload capacity of neighbor i[4]. We use $I^t_{i,n} \in \{0,1\}$ to indicate whether or not peer n obtains a non-zero download rate from neighbor i during time slot t, i.e., $I^t_{i,n} = 1$ indicates that the download rate is greater than zero. Then, the average download rate obtained by n prior to time slot t from neighbor i is defined as:

$$A^t_{i,n} = \left(\sum_{k=1}^{t-1} D^k_{i,n}\right) / \left(\sum_{k=1}^{t-1} I^k_{i,n}\right). \quad (5)$$

We choose to start releasing peers with the highest download rates because their high contributions provide greater benefits to other peers. In contrast, if we release peers with the lowest download rates, it is less likely that those donations could be useful to low capacity peers as they will have lower upload capacities and fewer useful blocks. The details of our modified mechanism are given in Algorithm 1. The last row of Table III also demonstrates the resulting "shift" in download capacity allocation (with our modified approach from one of our experiments). As a result of this approach (see details in Section V), no peer obtains an (un-necessarily) high download rate, but many peers have an opportunity to obtain download rates sufficient for playback.

C. The Reward Function for Advertisements

As discussed in Section III, appropriate duration of ads is a good incentive to motivate peers to continue contributing upload capacity while releasing unneeded download capacity. Specifically, in our approach, given a default ad duration that should be viewed by a peer, the length of peers' ads can be reduced as a reward for helping their neighbors or increased as a penalty for being helped by their neighbors. The mechanism used to compute rewards and penalties is detailed next.

Rewards. As described in Sections IV-B, a peer helps its neighbors by releasing neighbors, thus allowing them to obtain higher download rates and maintain quality of service. A peer's reward is computed based on the amount of help provided through this mechanism to its neighbors. We say peer n releases R^t_n neighbors in time slot t. If peer n did not release neighbor i, the expected download rate received from

[4]A download rate of zero might just mean that two peers did not share blocks during a particular time slot.

ALGORITHM 2: Penalty for peer n at time t

Peer n has m_n neighbors, and its upload rate to one neighbor is U_n;
The average download rates of neighbors are $A^t_{1,n}...A^t_{m_n,n}$;
The download rates in time slot t are $D^t_{1,n}...D^t_{m_n,n}$;
for *Neighbor from* $i = 1 \rightarrow m_n$ **do**
  if $D^t_{i,n} > 0$ *and* $A^t_{i,n} > \alpha * U_n$ **then**
    $P^t_n += D^t_{i,n} * (1 - 1/m_n)$
  end
end

i can be estimated as $A^t_{i,n}$, which is the historical average download rate from i. Thus, as in Algorithm 1, where peer n sorts neighbors (from high to low) based on their historical average download rates and releases the first R^t_n neighbors, the donation from peer n (or the reward that peer n could obtain) at time t can be computed as:

$$W^t_n = \sum_{i=1}^{R^t_n} A^t_{i,n}. \quad (6)$$

Penalties. A peer's penalty corresponds to the download rates a peer obtains as a result of its neighbors' "donations". Given the decentralized nature of our system, it is difficult to determine how much a neighbor's donation eventually increases a peer's download rate. Therefore, we use peers' local information to estimate this benefit, based on an expectation of how much download capacity a peer would not have obtained without our approach, as detailed next.

As we discussed in Section IV-B, peers form a cluster with neighbors that have similar capacities and consequently seldom exchange blocks with peers from other clusters, other than through optimistic unchoking. Thus, we treat the download rates obtained by peer n from peers in higher capacity clusters, other than those obtained through random selections, as being obtained due to "donations" made by higher capacity peers. Consequently, to determine the amount of download rates obtained through "donations", we need to determine how much is obtained through optimistic unchoking. To this end, we approximate the probability of a peer being chosen (to receive data) by neighbor i through the random selection by $\frac{1}{m_i - 4} \approx \frac{1}{m_i}$, where m_i is the number of peer i's neighbors. Since peer n does not know how many neighbors peer i has, we make the simplifying assumption that all peers have a similar number of neighbors; that is, in Algorithm 2, we set $m_i = m_n, \forall i$. Thus, if peer n obtains a download rate of $D^t_{i,n}$ from a higher capacity neighbor i in time slot t, then we estimate the "donated" download rate (from i to n) as being $D^t_{i,n} * (1 - \frac{1}{m_n})$.

What remains (before we can characterize the overall penalty) is to determine an appropriate cluster for each peer. We do this based on historical data, i.e., the average download rate, $A^t_{i,n}$, of neighbor i. Specifically, given that peer n's average upload rate to a neighbor is U_n, we consider neighbor i to be in the same cluster with peer n if $A^t_{i,n} \in [U_n/\alpha, \alpha * U_n]$, where α is a scaling parameter. Since a peer only receives "donations" from neighbors in clusters with higher capacities, we only account for a "donation" when it comes from a peer with a download rate higher than $\alpha * U_n$. Thus, the summation of the average download rates from higher capacity peers is P^t_n, as detailed in Algorithm 2. In our experiments, we set $\alpha = 1.5$ as the high upload capacities are at least 1.5 times higher than the low upload capacities, as listed in Table I. (In real systems, service providers can adjust this value according to their users' bandwidth capabilities as typical users are unlikely to change their upload bandwidth frequently.)

The duration of ads. Once we determine the reward and the penalty, we combine them in computing the duration of ads a peer should view. Given an ad period, to be viewed after every I time slots of content, the content provider determines the default total ad duration L_D and divides it equally among the ad periods. According to the contract, the content provider has a lower bound L_m and an upper bound L_M for the total ad duration to be viewed by a peer. Thus, if there are L ad periods, the actual ad duration in the j-th period to be viewed by peer n, L_n^j, is determined as follows:

$$L_n^j = \frac{1}{L} \min \left(\max \left(L_D - \lambda * \frac{1}{I} \sum_{k=t_j-I}^{t_j} (W_n^k - P_n^k), L_m \right), L_M \right), \tag{7}$$

where t_j is the start time of the j-th ad period.

In Section III, we discuss that the pair of parameters ($L_D = 6$ and $\lambda = 0.134$) provides the best market efficiency. Therefore, here, we adopt this pair of parameters in Eq. (7) and test if those parameters could enable our modifications to provide satisfactory service to peers, the content provider, and the advertisement provider. The corresponding detailed simulation study is described in Section V.

We note that our mechanisms for computing rewards and penalties require only local information. Therefore, our mechanisms do not require the use of central servers or information exchange between peers, which are needed in [12].

V. Evaluation

Our goal in this section is to evaluate the utility of ASPECT. To this end, we perform simulation-based experiments, in a controlled environment, in order to demonstrate the characteristics of our mechanisms and gain insight and understanding of the corresponding system.

We now describe how we implement the baseline approach (i.e. the original BitTorrent-like video streaming system discussed in IV-A) in our simulation environment as well as how we integrate our modified mechanisms (as described in Section IV-B) into this baseline approach. Briefly, we implement all peer client and server functionalities where simulated peers actually exchange information and data (as real peers do). However, since this is a simulated environment, peers do not actually play the videos, but only check that the required content is in the buffers when it is needed. A video pause is detected if it happens that a required block is not in the buffer. A summary of default parameter settings used here is given in Table IV.

In the simulations, after joining the system, a peer will re-registers itself and retrieves a set of random peers (at most 50 in our system) every 200 seconds from the tracker. In the BitTorrent protocol, a peer can only choke and unchoke peers once every 10 seconds to avoid oscillations. Therefore, we adopt 10 seconds as the interval T for peer's actions. To update block information, peers exchange bit-maps of blocks they have with their neighbors at the beginning of every interval. Then, peer n sends request messages to its neighbors that have blocks peer n does not have. Peer n will unchoke 4 peers that have the highest download rates to peer n for the 10 seconds prior to the next round of bit-map exchanges. For optimistic unchoking, peer n also re-selects the (randomly) unchoked neighbors every 10 seconds. After peer n is unchoked, it chooses missed blocks to download. Recall that our focus is on reducing video pauses as caused by insufficient download rates. Hence, we adopt an existing block selection algorithm as described in [19]. In this basic block selection algorithm, peers request blocks to be used in the near future first, and

TABLE IV. The parameters used in the experiments

# of peers in the system (N)	500
Length of an action interval (T)	10 seconds
Recorded simulation time (T_s)	1800 seconds
Max Advertisement : real content	0.31 : 1
Video bit rate	500 kbps
Upload bandwidth of content server	10240 kbps
Blocks per second	4
Number of peers unchoked by the server : a peer	20 : 5
α (the value used to distinguish peers' groups)	1.5
L_D (the default duration of ads)	6 minutes
λ (the ratio for translating rate to ads)	0.134

then use the rarest-first approach for those blocks that will not be needed in the near future. For instance, if video playback time of peer n is at t_n seconds, then peer n will give blocks in the $[t_n, t_n + T * 2]$ interval higher priority. If no block in $[t_n, t_n + T * 2]$ period can be selected for download, peer n will use rarest-first algorithm to fetch blocks after time $t_n + T * 2$. The central video server acts like another peer, with the difference being that it can unchoke a greater number of peers due to having higher upload capacity. (We provide details below.) All these mechanisms continue/repeat until the end of a simulation run.

Our modifications follow a similar design. However, the modified Peer Request algorithm insures that high capacity peers stop sending block request messages to some of the other high capacity peers. The specifics of how (and how many) high capacity peers are released are described in Section IV-B. Finally, having peers "donate" to others (as described earlier) introduces different duration of ads viewed, as detailed below.

To make sure that video pauses are not due to not having sufficient overall resources, but are rather due to inappropriate allocation of those resources, we only focus on experimental settings where there is sufficient *total* upload bandwidth to satisfy the total download demand. In our experiments, we set the total number of peers to 500, based on traces from the PPLive Project [30]. Upload capacities of peers are drawn from the distribution given in Table I. A central video server, that originally provides the content, has an upload capacity of 10240 kbps in our experiments. We consider single-layer video with constant bit rate (CBR) encoding in our simulations. In our experiments, the video bit rate is set to 500 kbps, as measured in [2]. We do not consider variable bit rate (VBR) videos in our simulations because we want to understand how much reduction of video pauses comes from our modifications, not from the low download rate requirements at some times. To focus on the above stated goals, we mostly consider a single streaming rate in our experiments; however, we do address heterogeneous streaming rates in Section V-B below. Based on the video bit rate and the upload capacity of the central video server, we have the video server unchoke 20 peers simultaneously, where all the unchoked peers can obtain sufficient download rates for video playback directly from the video server. According to [31], the average video viewing time per a user's visit is more than 22 minutes. Thus, for simplicity of exposition, we assume that there is no peer churn in our 30-minute simulation period. Moreover, we have no free-riders in our experiments (as that is not the focus of this work). Given these settings, the *total* upload capacity is sufficient for all peers to view video playback smoothly if the upload resources are allocated properly.

According to [22], the length of current ads on TV is $\approx 31\%$ of real content. For instance, in our 30-minute simulation period, there is typically only $\approx$ 23 minutes of real content with $\approx$ 7 minutes of ads. So, we set the maximal ad duration (L_M) to 7 minutes, and the minimal required length (L_m) to 1.5 minutes. The default duration (L_D) and λ obtained from our market-based model are 6 minutes and 0.134 respectively.

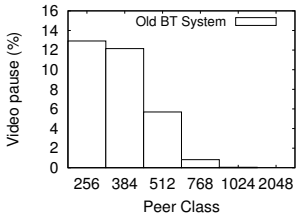

Fig. 7. Video pauses always happen on low capacity peers

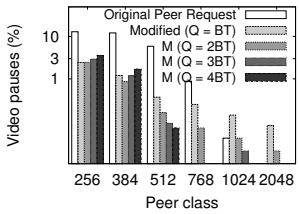

Fig. 8. Video pauses after applying our Peer Request mechanism

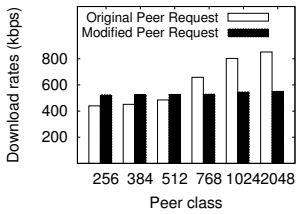

Fig. 9. The download rates of each class before and after applying our Peer Request mechanism

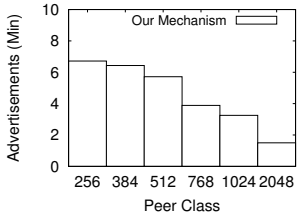

Fig. 10. The duration of ads for each class after our mechanisms

The primary evaluation metric, used in the remainder of this section, is the percentage of time the streamed video was paused, defined as follows. Given N peers in the system viewing video for length T_s, if the total length of video pauses experienced by peer i during this experiment is V_i, the percentage of video pauses for all N peers in the system is computed as:

$$\%\, of\, pause\, time = \sum_{i=1}^{N} V_i/(N * T_s) * 100\%. \quad (8)$$

In order not to have initial "warm-up period" results skew the outcome, we run each simulation for 2400 seconds, and only record the results from the last 1800 seconds, where each peer has already learned around $1/3$ of the peers in the system and peers already form their clusters. Moreover, most simulation results presented in this section are obtained with $95\% \pm 5\%$ confidence intervals[5]. Due to lack of space, we only show a subset of our results in what follows; however, results for other settings are qualitatively similar.

A. Buffer Starvation

In this experiment, we show peers with lower capacity experience frequent video pauses. We simulate the baseline approach, i.e., an original BitTorrent-like video streaming system, and record the percentage of video pauses of different classes, i.e., as in Eq. (8) but on a per class basis, as shown in Fig. 7. For instance, low capacity peers experience more than 3 minutes of video pauses in a 30-minute video (such as a typical TV show), particularly peers in the 256 kbps class, who experience almost 4 minutes of video pauses. On the other hand, high capacity peers seldom experience a video pause. This experiment demonstrates that buffer starvation (and subsequent video pauses) is potentially a significant problem in P2P-based video streaming systems.

B. Peer Request Modification

As shown in Section V-A, the video playback quality of low capacity peers is affected by the high frequency of video pauses. In this experiment, we demonstrate that our proposed Peer Request mechanism can significantly reduce video pauses experienced by low capacity peers. We compare the percentage of video pauses experienced by the different classes of peers under the original Peer Request mechanism with those under our modified version; the results are depicted in Fig. 8. As noted in Section IV-B, a peer starts to release its neighbors after the number of continuous blocks cached in its buffer, Q, is larger than a threshold. Therefore, in Fig. 8, we also demonstrate the performance of our modified versions when different thresholds are used. Here, we observe that our mechanism reduces video pauses under all thresholds used in our experiments. Generally, our mechanism can significantly reduce video pauses of peers in the 256 kbps class (i.e., to reduce more than half of pauses experienced under the original BitTorrent-like approach). Moreover, video pauses of peers in the 384 kbps class are nearly eliminated (i.e., only around 1 percent of video pauses are left). However, the performance of our mechanism highly depends on the threshold, particularly when the threshold is very small. As expected, a smaller threshold increases peers' probability of releasing neighbors and increases the download rate of low capacity peers; however, it makes peers more sensitive to the download rate changes, resulting in a higher probability of experiencing video pauses. On the other hand, a higher threshold enables peers to absorb the change in download rates, but makes low capacity peers not being likely to obtain donations from high capacity peers. Therefore, in our work, we choose a reasonably conservative threshold of twice required blocks for each time period.

To demonstrate that the video pauses are reduced through the "donation" of download capacity by high capacity peers, in Fig. 9, we depict the download rates obtained by peers in different classes. This figure shows that low capacity peers can increase their download rates, once some of the capacity is "released" by high capacity peers. For instance, peers in the 256 kbps class obtain nearly sufficient download rates for video playback, thus resulting in significantly fewer video pauses. (Recall that the video bit rate is 500 kbps.)

Although in our experiments only single-layer video is used, our approach can be easily extended for systems with heterogeneous streaming rates through the use of layered video coding. The main difference in systems with heterogeneous streaming rates is that high capacity peers may prefer to use their "un-needed" capacity to view higher rate (and hence quality) videos rather than release it in order to reduce the duration of ads viewing. However, as measured in [32], the highest bit rate used in current video streaming systems is 9 Mbps, which is smaller than the rate of many cable Internet connections. Hence, as long as high capacity peers are interested in viewing shorter duration ads, low capacity peers can still obtain increased download rates from high capacity peers through our mechanism.

C. The Duration of Advertisements

In this experiment, we use the parameters L_D and λ, determined in Section III, to adjust the duration of ads (see Section IV-C). We record the duration of ads viewed by different classes of peers. As shown in Fig. 10, lower capacity peers view more than the default length, which is 6 minutes, while higher capacity peers view significantly shorter duration ads. However, since the increased/decreased duration of ads is proportional to the expected download rate change (refer to Eq. (6) and Algorithm 2), peers will have significant differences in their ad durations (e.g., peers in 2048 kbps class always release peers with highest capacities, resulting in significantly

[5]Results for the 2048 kbps class are obtained with lower confidence - video pauses rarely occur for that class, and hence, it is difficult to obtain simulation results with high confidence intervals.

decreased ad durations). Compared to the results in Fig. 3, which are obtained by simulating with arbitrary parameters, the parameters from our market-based model deliver the most differentiated ad durations without violating constrains (i.e., these durations are between L_m and L_M). Therefore, with ASPECT, the content provider can still satisfy requirements from ad providers (i.e., delivering the minimal duration of ads), while incentivizing peers to contribute resources by offering differentiated ad durations to peers, based on their resource contributions as well as the amount of resources they receive.

VI. Conclusions

We proposed an Ad-Driven Streaming P2P Ecosystems (ASPECT) in this paper, in the context of P2P video streaming systems, for peers to donate their (un-needed) download capacity, in order to improve overall QoS in the system. To support such "re-allocation" of resources, we proposed a modified Peer Request mechanism, which facilitates donation (by peers) of potentially available download capacity. To provide appropriate incentives, we viewed the P2P-based video streaming system as a market-based model, which encourages peers to release their download rates for shorter ad durations. Our simulation-based experiments demonstrated that ASPECT can significantly reduce video pauses, thus increasing QoS, while maintaining overall commitment to the ad provider.

In this paper, we focused on the block exchange progress; thus, we only considered a single channel (i.e., users sharing the same video) and used a simple block selection mechanisms designed for video streaming. However, our work can be easily combined with other block selection mechanisms as well as extended to a multi-channel scenario. Moreover, our market-based model can be extended to multiple content providers and ad providers with various requirements in the market.

References

[1] R. Kumar, Y. Liu, and K. Ross, "Stochastic fluid theory for p2p streaming systems," in *INFOCOM*. IEEE, 2007.

[2] X. Hei, C. Liang, J. Liang, Y. Liu, and K. Ross, "A measurement study of a large-scale p2p iptv system," *Transactions on Multimedia*, 2007.

[3] Y. Xu, E. Altman, R. El-Azouzi, M. Haddad, S. Elayoubi, and T. Jimenez, "Probabilistic analysis of buffer starvation in markovian queues," in *INFOCOM*, 2012.

[4] Y. Xu, E. Altman, R. El-Azouzi, S. Elayoubi, and M. Haddad, "Qoe analysis of media streaming in wireless data networks," *NETWORKING*, 2012.

[5] A. ParandehGheibi, M. Médard, A. Ozdaglar, and S. Shakkottai, "Avoiding interruptions a qoe reliability function for streaming media applications," *IEEE JSAC*, 2011.

[6] Z. Liu, Y. Shen, S. S. Panwar, K. W. Ross, and Y. Wang, "Using layered video to provide incentives in p2p live streaming," in *Proceedings of the 2007 workshop on Peer-to-peer streaming and IP-TV*. ACM, 2007.

[7] J. Mol, D. Epema, and H. Sips, "The orchard algorithm: P2p multicasting without free-riding," in *Peer-to-Peer Computing*. IEEE, 2006.

[8] J. Rückert, O. Abboud, T. Zinner, R. Steinmetz, and D. Hausheer, "Quality adaptation in p2p video streaming based on objective qoe metrics," *NETWORKING*, 2012.

[9] Hulu, "Hulu," http://www.hulu.com, 2014.

[10] "comscore releases december 2013 u.s. online video rankings," http://www.comscore.com/Insights/Press-Releases/2014/1/comScore-Releases-December-2013-US-Online-Video-Rankings.

[11] "Hulu:15 or 30 Video Commercial," http://www.hulu.com/advertising/ad-product/video/15-or-30-video-commercial/, 2014.

[12] B. Wang, A. Chow, and L. Golubchik, "P2p streaming: use of advertisements as incentives," in *Multimedia Systems*. ACM, 2012.

[13] P. Hoong and H. Matsuo, "Push-pull incentive-based p2p live media streaming system," *WSEAS transactions on communications*, 2008.

[14] Z. Liu, Y. Shen, K. Ross, S. Panwar, and Y. Wang, "Substream trading: Towards an open p2p live streaming system," in *ICNP*, 2008.

[15] T. Qiu, I. Nikolaidis, and F. Li, "On the design of incentive-aware p2p streaming," *Journal of Internet Engineering*, 2007.

[16] N. Parvez, C. Williamson, A. Mahanti, and N. Carlsson, "Analysis of bittorrent-like protocols for on-demand stored media streaming," in *ACM SIGMETRICS Performance Evaluation Review*. ACM, 2008.

[17] Y. Yang, A. Chow, L. Golubchik, and D. Bragg, "Improving qos in bittorrent-like vod systems," in *INFOCOM*. IEEE, 2010.

[18] L. D'Acunto, N. Andrade, J. Pouwelse, and H. Sips, "Peer selection strategies for improved qos in heterogeneous bittorrent-like vod systems," in *Multimedia (ISM)*. IEEE, 2010.

[19] A. Vlavianos, M. Iliofotou, and M. Faloutsos, "Bitos: Enhancing bittorrent for supporting streaming applications," in *INFOCOM*, 2006.

[20] Y. R. Choe, D. L. Schuff, J. M. Dyaberi, and V. S. Pai, "Improving vod server efficiency with bittorrent," in *Multimedia*. ACM, 2007.

[21] D. Wu, Y. Liang, J. He, and X. Hei, "Balancing performance and fairness in p2p live video systems," *Circuits and Systems for Video Technology, IEEE Transactions on*, 2013.

[22] W. Schmidt, "How Much TV Commercial Length has Grown over the Years," http://www.waynesthisandthat.com/commerciallength.htm, 2014, [Online; accessed 20-June-2014].

[23] C. W. Cobb and P. H. Douglas, "A theory of production," *The American Economic Review*, pp. 139–165, 1928.

[24] A. Mas-Colell, M. D. Whinston, J. R. Green *et al.*, *Microeconomic theory*. Oxford university press New York, 1995, vol. 1.

[25] M. Dischinger, A. Haeberlen, K. Gummadi, and S. Saroiu, "Characterizing residential broadband networks," in *IMC, ACM*, 2007.

[26] X. Zhang, J. Liu, B. Li, and T.-S. P. Yum, "Coolstreaming/donet: a data-driven overlay network for peer-to-peer live media streaming," in *INFOCOM*. IEEE, 2005.

[27] G. Huang, "Pplive: A practical p2p live system with huge amount of users," in *Proceedings of the ACM SIGCOMM Workshop on Peer-to-Peer Streaming and IPTV Workshop*, 2007.

[28] B. Q. Zhao, J. Lui, and D.-M. Chiu, "Exploring the optimal chunk selection policy for data-driven p2p streaming systems," in *P2P*, 2009.

[29] A. Legout, N. Liogkas, E. Kohler, and L. Zhang, "Clustering and sharing incentives in bittorrent systems," in *SIGMETRICS Performance Evaluation Review*. ACM, 2007.

[30] L. Vu, "Pplive project," 2008. [Online]. Available: http://cairo.cs.uiuc.edu/longvu2/pplive.html

[31] S. S. Krishnan and R. K. Sitaraman, "Video stream quality impacts viewer behavior: inferring causality using quasi-experimental designs," in *Internet measurement conference*. ACM, 2012.

[32] T.-Y. Huang, N. Handigol, B. Heller, N. McKeown, and R. Johari, "Confused, timid, and unstable: picking a video streaming rate is hard," in *Internet measurement conference*. ACM, 2012.