Can Accurate Predictions Improve Video Streaming in Cellular Networks?

Xuan Kelvin Zou[1], Jeffrey Erman[2], Vijay Gopalakrishnan[2], Emir Halepovic[2], Rittwik Jana[2], Xin Jin[1], Jennifer Rexford[1], and Rakesh K. Sinha[2]

[1] Department of Computer Science, Princeton University, Princeton, NJ
[2] AT&T Labs – Research, Bedminster, NJ
[1]{xuanz, xinjin, jrex}@cs.princeton.edu
[2]{erman, gvijay, emir, rjana, sinha}@research.att.com

ABSTRACT

Existing video streaming algorithms use various estimation approaches to infer the inherently variable bandwidth in cellular networks, which often leads to reduced quality of experience (QoE). We ask the question: "If accurate bandwidth prediction were possible in a cellular network, how much can we improve video QoE?". Assuming we know the bandwidth for the entire video session, we show that existing streaming algorithms only achieve between 69%-86% of optimal quality. Since such knowledge may be impractical, we study algorithms that know the available bandwidth for a few seconds into the future. We observe that prediction alone is not sufficient and can in fact lead to degraded QoE. However, when combined with rate stabilization functions, prediction outperforms existing algorithms and reduces the gap with optimal to 4%. Our results lead us to believe that cellular operators and content providers can tremendously improve video QoE by predicting available bandwidth and sharing it through APIs.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed systems-Distributed applications; C.4 [**Performance of Systems**]: Modeling techniques-Measurement techniques

Keywords

Video streaming; HTTP; DASH; Adaptation; Algorithm; Design; Performance; Prediction; Cellular

1. INTRODUCTION

Video streaming accounted for 66% of total Internet traffic [5] and accounts for over 40% of cellular traffic [7]. This demand has forced cellular providers to deploy significant capacity to support high quality video streaming. Yet, despite these efforts, achieving reliable video streaming over cellular networks has proven to be difficult. For example, it is reported that the fraction of stalled videos increases with video quality, with 10.5% of 240p videos stalling while 45.7% of 720p videos experiencing a stall [6].

Most content providers today use adaptive-bit-rate (ABR) streaming where the goal is to match the delivery rate of "chunks" of the video to available end-to-end bandwidth. While simple in principle, practical implementations have to *infer* the available bandwidth and adjust rates for chunks while balancing metrics like quality, interruptions, number of rate switches (i.e., rate stability). State-of-the-art algorithms differ in how they infer available bandwidth; while some use historical throughput [10, 12], others use buffer occupancy [8] or traffic shaping [4].

Accurate inference of available bandwidth is non-trivial in part due to varying link capacities, congestion, and other factors. This task is particularly challenging in cellular networks due to the inherent variability in signal strength, interference, noise, and user mobility — all of which cause the bandwidth to vary widely over time. Consequently, there is a significant mismatch between estimates used by existing algorithms and the actual available network bandwidth, which results in low quality of experience (QoE) over cellular networks. However, recent works open a promising possibility to accurately predict available bandwidth at short and medium time scales [14, 11].

In this paper, we focus on the following question: *If accurate bandwidth prediction were possible in a cellular network, how much can we improve video QoE*? We consider cellular networks because they present both a challenge and an opportunity. While it is challenging due to the high link variability, the cellular radio link is carefully scheduled. A base station tracks multiple network metrics (routinely used in the scheduling process) that can be used to predict available bandwidth in near future (several seconds). In addition, the architecture of the network allows an operator to have a view of all devices, their link statistics, radio spectrum availability and instantaneous traffic demand. Our goal is to exploit this network information to derive available bandwidth and expose it to content providers through APIs. We further focus on video durations on the order of several minutes, which is most common in cellular networks [7].

To that end, we first identify the gap between existing algorithms (e.g., FESTIVE [10], BBA [8]) and the optimal by formulating a Mixed Integer Linear Program (MILP).

HotMobile'15, February 12–13 2015, Santa Fe, NM, USA.

http://dx.doi.org/10.1145/2699343.2699359.

Since existing algorithms are not designed to use bandwidth predictions, we develop a class of Prediction-Based Adaptation (PBA) algorithms that use the predicted bandwidth.

Our key findings can be summarized as follows:

- Existing algorithms fail to fully utilize available bandwidth, achieving only 69%-86% of optimal quality. The performance gap is pronounced during startup where existing algorithms achieve only 15%-20% of optimal in the first 32 seconds, and 22%-38% of optimal in the first 64 seconds.
- Naive algorithms that utilize *only* predicted bandwidth for chunk rate selection do not work well and cause numerous and erratic quality switches.
- PBA that combines short-term predictions (e.g., one chunk duration) with buffer occupancy and/or rate stability function outperforms existing algorithms, achieving nearly 96% of optimal quality during startup and over the entire video, thereby outperforming existing algorithms by up to ∼40%.
- PBA algorithms with different stability functions can trade off stalls and stability while maintaining much improved average quality.

Our results lead us to believe that there is tremendous improvement to be had in video QoE by accurately predicting available bandwidth and exposing them to content providers through APIs.

2. BACKGROUND AND MOTIVATION

The available bandwidth in both wired and wireless networks varies over time. To overcome this variability, the industry has shifted towards ABR streaming to deliver videos over the Internet. ABR streaming is a client-driven approach in which a video client tries to match the delivery rate of the video to that of the available end-to-end bandwidth. To make this feasible, ABR streaming breaks videos into "chunks" of a few seconds (typically 2-10 sec) and encodes each chunk at multiple bit rates, representing different levels of quality. Multiple encoding bit rates also directly correlate to PSNR levels resulting from compressing the video signal [3]. The client's task is to choose chunks of the "correct" encoding rate.

Depending on the perceived network conditions, playout buffer or other criteria, clients attempt to optimize various metrics that comprise users' QoE, including quality, interruptions and stability. For example, a client can switch to a low-quality version of the video to avoid buffer underflow during temporary network congestion, and switch back to higher quality after network conditions improve. Variants of this technology have been widely deployed in commercial systems, including Netflix, Microsoft Smooth Streaming, Apple HTTP Live Streaming (HLS) and Dynamic Adaptive Streaming over HTTP (DASH).

While simple in principle, practical client implementations of ABR streaming have to *infer* the available bandwidth and adjust chunk encoding rates accordingly. State-of-the-art approaches typically use past observations to estimate available bandwidth. One representative is FESTIVE [10], an adaptation algorithm that includes features designed to provide fair and stable performance even when multiple ABR video players compete for bandwidth. Among other features, FESTIVE uses harmonic mean of previous chunk throughputs and includes a stability function that delays video rate updates to minimize rate switches.

In contrast, Buffer-Based Adaptation (BBA) [8] is proposed to dispense with historical averaging and solely use buffer level to drive selection of video rates. The rationale is that buffer level indirectly reflects the historical throughput, which obviates a need for direct measurement and estimation. However, the authors concede that when the buffer is empty or very low, such as during the startup phase, historical bandwidth estimation is necessary. Prior work compares buffer-based and rate-based streaming algorithms to suggest that bandwidth prediction may be beneficial [15].

We contend that the *inference* of available bandwidth leads to innacurate estimates of actual bandwidth, resulting in low video QoE over cellular networks. Compared to prior work that used synthetic traces and lacked prediction-based adaptation [15], we design prediction-based class of algorithms and use real traces. We study the benefits of bandwidth prediction for potentialy complementing or even replacing existing algorithms to improve video QoE.

Table 1: Optimization variables and parameters

Symbol	Meaning
Q	number of video quality levels (play bit rates)
R_i^q	size of the i-th chunk at the q-th quality level
x_i^q	binary variable indicating selection of R_i^q
C_i	predicted bandwidth for the i-th chunk duration
L_i^j	portion of i-th chunk downloaded in duration j
M	maximum buffer size (in multiples of D)

3. UPPER BOUND ON QUALITY

We divide the video session into time intervals of length equal to chunk duration and assume knowledge of the available average bandwidth during each chunk duration.

We seek to understand the highest possible quality for this session such that we have no interruptions. While this is clearly an idealized and impractical setting, it gives us an upper bound on the video quality possible and allows us to quantify how well existing adaptation algorithms are doing.

We formulate this optimal rate selection as a Mixed Integer Linear Program (MILP). Table 1 lists the variables and parameters used in our formulation.

For the i-th video chunk, the optimization selects exactly one quality for playback. In other words, the indicator variable x_i^q is one for exactly one quality level q and the chunk size is given by

$$ChunkSize(i) = \sum_{q=1}^{Q} R_i^q * x_i^q.$$

For each video chunk i, the optimization needs to find its quality level (x_i^q) and the goal is to maximize

$$\sum_i ChunkSize(i) = \sum_{i,q} R_i^q * x_i^q,$$

subject to constraints defined below.

3.1 Constraints

Unique quality: We pick exactly one quality for each chunk:

$$\forall i,\ \sum_{q=1}^{Q} x_i^q = 1.$$

No stalls: To avoid interruptions, the i-th chunk needs to finish downloading by the end of chunk duration i to be played out in chunk duration $i+1$. In other words, we can not download any portion of the i-th chunk in any chunk duration indexed higher than i:

$$\forall i, j > i,\ L_i^j = 0.$$

Limited buffer: We do not want to download too many chunks ahead of their play time because, in case of abandonment, these chunks represent wasted network bandwidth. For this reason, most practical implementations have a limited buffer size. We capture this by restricting that a chunk can *not* get downloaded M chunk durations ahead of its play time:

$$\forall i, j \leq i - M,\ L_i^j = 0.$$

Bandwidth availability: The predicted bandwidth in the j-th chunk duration, C_j, should be sufficient to download all (portions of) chunks downloaded in the j-th duration:

$$\forall j,\ \textstyle\sum_i L_i^j \leq C_j.$$

Download consistency: Portions of i-th chunk downloaded in various chunk durations has to sum up to the size of the i-th chunk:

$$\forall i,\ \textstyle\sum_j L_i^j \geq \sum_{i,q} R_i^q * x_i^q.$$

We note that we do not have any constraints to force download of chunks in order and the solution may interleave the downloading of video chunks (e.g., download chunks 1 and 3 in the 1st chunk duration and then download chunk 2 in the 2nd chunk duration). We can prove (omitted for brevity) that given such a schedule, we can transform it into a sensible schedule where chunks are downloaded in sequence without violating any of the constraints.

3.2 Data and Evaluation

We compare state of the art adaptation algorithms to the MILP using 20 4G/LTE cellular traces that we collected from several large US cellular providers. Each trace provides the per-second available bandwidth over 360 seconds. To accurately represent some of the more challenging conditions to deliver ABR video, the traces were obtained from various locations with different terrains (highways, local roads, etc.) and represent situations where bandwidth is more constrained and fluctuating quickly due to changing cellular conditions. The average bandwidth across traces is 6.5 Mbps.

We assume that each chunk is 4 seconds long and a maximum buffer of 64 seconds (or equivalently, 16 chunks). We take 10 different video encoding rates from one major content provider: 235, 375, 560, 750, 1050, 1750, 2350, 3000, 3850, and 4300 Kbps [8] and assume constant bit rate for each chunk. The selected chunk duration and buffer size approximate common settings of several content providers for wired and wireless environments. A few traces could not satisfy MILP's strict no-stall constraint. We solved MILP on the remaining traces using a commercial solver and compared the average bit rate to those of FESTIVE and BBA[1].

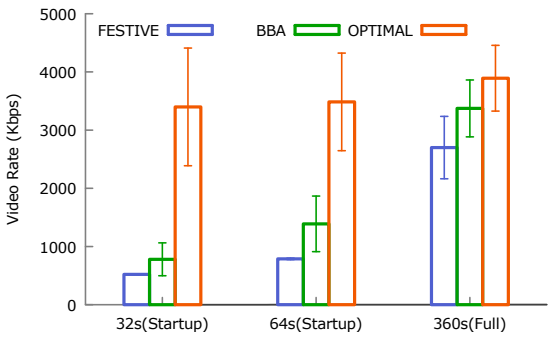

Figure 1: FESTIVE and BBA fall significantly short of optimal video quality.

Over the full traces of 360 sec (90 chunks), on average, FESTIVE and BBA achieve 68.6% and 85.7% of optimal, respectively (Figure 1). Using a re-run of MILP to maximize the rate during short and long start-up phases of 32 and 64 seconds, we find that the achievable gains obtained by MILP are significantly higher. FESTIVE and BBA reach only 15.0% and 20.1% of optimal for 32 sec, and 21.8% and 33.4% for 64 sec, respectively. These results clearly show that significant benefits can be gained by prediction, especially during startup.

4. PREDICTION-BASED ADAPTATION

In this section, we develop online versions of PBA by relaxing the assumption of perfect bandwidth knowledge for the entire session duration to shorter prediction horizons. We first highlight a naive PBA which selects the next chunk's quality based only on predicted bandwidth. This turns out to have unintended consequences.

Then, we develop a more sophisticated online PBA algorithm that obtains significant improvements in quality by combining available bandwidth prediction for only the next few seconds (e.g., one chunk duration) with buffer occupancy or rate stability function.

4.1 Naive PBA

We motivate our algorithm with a naive solution: choose the highest bit rate that is less than the predicted bandwidth. The prediction horizons are short (1 chunk duration of 4 seconds), medium (5 chunks), and long (10 chunks). The client obtains a prediction just before each chunk download. We assume that the future available bandwidth is known for a fixed time period and represented by a single average value.

Evaluation: Figures 2-4 show the three behaviors for three prediction horizons. The video rate selection based on 1-chunk horizon closely follows predicted bandwidth, leading to numerous (39) and erratic quality switches (Figure 2). This greedy rate selection is clearly reflected in very slow buffer filling, and no use of the buffer to stabilize quality.

[1]We implement full FESTIVE with recommended stability parameters [10] and BBA-2 variant [8].

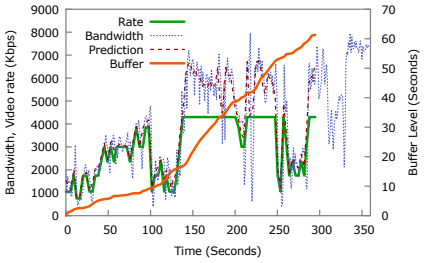

Figure 2: 1-chunk lookahead causes numerous and erratic rate switches.

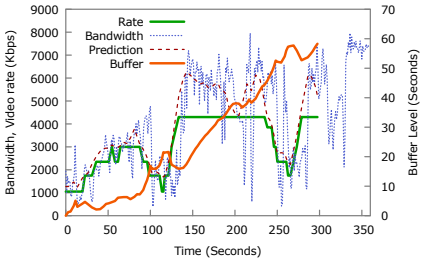

Figure 3: 5-chunk lookahead has fewer rate switches but exhibits poor buffer usage.

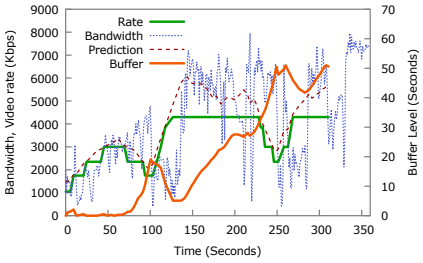

Figure 4: 10-chunk lookahead does not consider variability and leads to stalls.

Figure 3 shows a 5-chunk horizon (20 seconds), which stabilizes quality (18 switches) by averaging predicted bandwidth over a longer period, and leads to better buffer use. However, in some instances, the buffer is not appropriately used, i.e. it is refilled while bandwidth has dropped (around 120 s), and drained while bandwidth is increasing (around 40 seconds).

Finally, Figure 4 shows another extreme by taking a 10-chunk horizon (40 seconds), further stabilizing quality to 14 switches. However, this causes stalls (a cumulative stall time of 7.2 sec), because the long-term average does not include information about bandwidth variability and outages within the prediction horizon. In addition, buffer level is not considered during startup to recognize the danger of underflow due to high bandwidth variability.

While the average quality between three scenarios is similar (3.05 to 3.34 Mbps), the QoE differs significantly. There are several shortcomings of the sole use of prediction to determine quality: (i) stability is heavily impacted by highly fluctuating bandwidth, (ii) buffer fills slowly risking underflow because most of the bandwidth is used to maximize quality, and (iii) even when buffer level is high, it may not be used appropriately with respect to bandwidth fluctuations.

We take these shortcomings as takeaways to drive the design of prediction-based adaptation algorithm that improves video QoE, by considering stability, stalls, and buffer occupancy.

Algorithm 1: PBA Rate Selection

Input: B: Buffer occupancy
C: Predicted available bandwidth
R_{last}: Last downloaded video rate, initialized to the highest rate

Output: R_{next}: Video rate for next chunk

if $B > B_{max} - D$ **then**
  Sleep for $B + D - B_{max}$ seconds
$ref = \max\{i : R_i \leq C\}$
if $B \leq B_{risky}$ **then**
  $ref = \max\{ref - 1, 1\}$
  if $R_{ref} < R_{last}$ **then**
    $R_{next} = \max\left(\{R : \frac{B}{D} + \frac{C}{R} - 1 > 2\} \cup \{R_1\}\right)$
  else
    $R_{next} = R_{ref}$
else if $B \geq B_{safe}$ **then**
  $R_{next} = \max\{R_{ref}, R_{last}\}$
else
  if $R_{ref} \leq R_{last}$ **then**
    $R_{next} = R_{last}$
  else
    $\Delta B = D * (C/R_{ref} - 1)$
    $B_{empty} = B_{max} - B$
    if $\Delta B > 0.15 * B_{empty}$ **then**
      $R_{next} = R_{ref}$
    else
      $R_{next} = R_{ref-1}$

4.2 PBA

Since existing adaptation algorithms are not designed to use prediction, we first design a new PBA algorithm. Our PBA uses prediction, explicitly considers buffer occupancy and aggressively tries to stabilize rate selection.

We divide the buffer into three zones – *safe*, *transient*, and *risky* – based on two buffer occupancy thresholds, B_{safe} and B_{risky}. We set the safe and risky thresholds to 90% and 30% of buffer size B_{max}, respectively. A decision on the video rate for the next chunk, R_{next}, is made differently depending on the zone in which the current buffer occupancy B is, predicted available bandwidth C, and last video rate downloaded R_{last}. Algorithm 1 provides the pseudo-code.

The algorithm starts by checking the buffer occupancy against the target maximum B_{max}, and waits in case there is not enough space to store the next chunk of duration D, thereby protecting from buffer overrun. Once buffer space is available, the first step is to select a reference video rate, R_{ref}, to be the highest available rate below predicted bandwidth C.

In the risky zone, a conservative approach is used to prevent stalls and replenish the buffer, by first reducing R_{ref} by one level. If R_{ref} is higher than the rate of last chunk, R_{last}, we upswitch to R_{ref}. However, if R_{ref} is lower than R_{last}, this means that the reduction is recommended. To avoid sudden quality drop to R_{ref}, we pick the highest rate R without draining the buffer to less than two chunks, which performs better than one chunk when variable chunk sizes are used. B/D is the number of chunks in the buffer before requesting next chunk, C/R is the additional number of chunks that will be downloaded, while -1 represents one chunk that is expected to be played from the buffer while the next chunk is retrieved. During startup, with $B = 0$, this constraint selects the rate of the first chunk equal to 1/3rd of the predicted bandwidth. If such a rate cannot be found, the only decision that can be made is to select the lowest rate R_1.

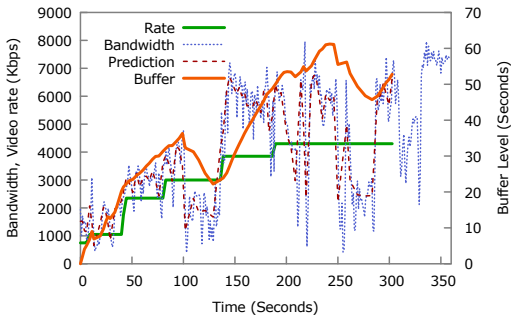

Figure 5: PBA with 1-chunk lookahead is stable and stall-free while offering high video quality.

In the safe zone, the algorithm aggressively selects the larger of R_{ref} and R_{last} to maintain stability and ride out short-term bandwidth variations.

In the transient zone, various approaches can be taken to balance the buffer occupancy, high quality, stability, and other objectives. In this implementation, we take a more aggressive approach. When R_{ref} is lower than R_{last}, we stick to R_{last} with the expectation that bandwidth will recover. If R_{ref} is higher than R_{last}, we want to grow the buffer. The algorithm will select rate R_{ref} if downloading at R_{ref} will allow at least 15% of the empty buffer space to be filled. Otherwise, we reduce by one rate and select R_{ref-1}.

Evaluation: Figure 5 shows for a specific trace how PBA behaves when buffer level is taken into consideration. Compared to naive PBA, the buffer is filled faster during startup, which is desirable. When the buffer level is moderate to high, it is aggressively exploited to ride out short fade durations and maintain high quality (100-140 seconds and 260-280 seconds). The average rate over 360 sec is 3.15 Mbps with no stalls and only 6 rate switches. Therefore, average quality is similar to naive PBA, but with significantly improved stability and appropriate buffer usage.

Next, we compare PBA to state-of-the-art proposals, FESTIVE and BBA. In addition to our PBA algorithm, we also implement an alternative using a known stability function employed by FESTIVE, called "delayed update". Using the aforementioned data set, we run simulations to investigate how close to optimal each adaptation algorithm gets. Table 2 lists algorithms and the percentage of optimal (MILP) they can achieve using a 64-second buffer. Recall that over the entire trace durations, FESTIVE and BBA come within 68.6% and 85.7% of optimal, respectively. PBA with buffer-based stability function (PBA-BB) reaches 95.8% of optimal quality, representing a relative improvement of ∼40% over FESTIVE. PBA with delayed update (PBA-DU) is within 91.4% of optimal quality. During startup phase (32 s), the value of prediction becomes abundantly clear, with up to 95.8% of optimal quality, while FESTIVE and BBA remain extremely sub-optimal.

Comparing metrics other than video quality shows a slight advantage of BBA because of fewer stalls. This is not surprising given that BBA is designed specifically to avoid stalls, unlike PBA and FESTIVE. While FESTIVE and BBA are nearly as stable as optimal, PBA-BB registers the best stability, and PBA-DU reduces stalls by decreasing stability, which is a very desirable trade-off [2]. PBA algorithms with different stability functions can trade off stalls and stability while maintaining much improved average quality.

Table 2: PBA achieves near-optimal quality.

	%Avg.rate 360 s	%Avg.rate 32 s	Number of stalls	%Rate switches
Optimal	100%	100%	0	100%
FESTIVE	68.6%	15.0%	34	101.3%
BBA	85.7%	20.1%	17	93.3%
PBA-BB	95.8%	84.8%	31	43.0%
PBA-DU	91.4%	95.8%	23	181.2%

We finally seek to quantify the benefit of prediction, i.e. to show that the benefits come from prediction and not from other parts of the adaptation algorithm. This is not a straight-forward task, since every algorithm is designed to work best with a specific bandwidth estimate. Nevertheless, we design a basic benchmark using our adaptation algorithm with two stability functions and swap predicted bandwidth with output of other heuristics. This approach maintains the while adaptation process the same except how reference rate is computed. We use harmonic mean of last 10 chunks and last chunk throughput as the two heuristics, and 4-second (1 chunk) look-ahead for PBA. Harmonic mean is the core feature of FESTIVE and last chunk throughput is used by BBA during startup.

Table 3 shows the relative improvement that prediction (PBA) offers compared to other heuristics for a given stability function. We make several interesting observations. First, when looking at the average bit rates over the entire trace of 360 sec, all algorithms do well. Prediction provides up to 4.8% improvement on average quality while maintaining overall better number of stalls and switching statistics. Second, during the "startup" phase, prediction offers nearly 18% improvement in video rate compared to harmonic mean. Finally, out of the three, last chunk strategy performs the worst in terms of stall and switch statistics. On the other hand, harmonic-based strategy has commensurate stall and switching statistics.

With "delayed update", prediction outperforms others by a similar margin during startup phase. Prediction offers significant reduction in the number of stalls and the cumulative time stalled. However, this comes at the expense of an increased number of switches. Reducing the number of stalls is a desirable feature in terms of customer engagement [2]. Finally, PBA is stall-free in 6 more traces than harmonic (a 25% improvement).

5. OPEN CHALLENGES

While our results show that accurate predictions of available network bandwidth improve the quality of video over cellular networks, generating accurate predictions is non-trivial. We discuss some of the challenges in generating bandwidth predictions in real time and conveying them at scale.

Generating accurate predictions: The cellular radio link is carefully scheduled. A base station tracks multiple network metrics and uses them in the scheduling process. In addition, the architecture of the network allows an operator to have a view of all devices, their link statistics, radio spectrum availability, mobility patterns, and instantaneous traffic demand. Moreover, recent work [14] shows that it is possible to get very accurate predictions (98% accuracy) for

Table 3: PBA improvement over heuristics based on Harmonic mean and Last chunk throughput.

	Avg. rate, 360 s		Avg. rate, 32 s		Avg. rate, 64 s		Time stalled		Number of stalls		Rate switches	
Stability function	Last	Har	Last	Har	Last	Har	Last	Har	Last	Har	Last	Har
Buffer-based	0.4%	4.8%	10.0%	17.9%	4.6%	13.8%	35.2%	-12.3%	22.6%	0.0%	34.8%	-11.9%
Delayed update	-1.7%	0.0%	11.3%	17.3%	4.3%	11.8%	132.9%	184.5%	69.6%	139.1%	10.0%	-63.6%

short time periods (500 msec) just by observing network performance at a stationary client device. There is also evidence that we can obtain good mobility models [13] and generate feasible throughput predictions [11] over short time periods (5 seconds) even when users are mobile. These results lead us to believe that it is feasible for cellular operators to predict available bandwidth accurately.

However, the exact mechanisms, data and algorithms that need to be combined to generate accurate, real-time predictions of available bandwidth are still unknown. Further, it would be desirable to have long term predictions (e.g., MILP is able to significantly increase quality), and have multiple predictions (a very accurate short term prediction and an indicative long term prediction) or even a distribution of bandwidth over the predicted period. We are pursuing these questions as part of our ongoing work.

Exposing computed predictions: Assuming these predictions can be computed, exposing them to client applications in a timely manner poses interesting systems and protocol challenges. One proposal considers a throughput guidance protocol where bandwidth information is embedded in TCP headers [1]. This approach makes the information available to TCP stacks of senders, to be potentially used by all applications. The evaluation based on simulation suggests benefits for video streaming similar to our results presented in this paper.

We envision that the cellular provider can expose bandwidth predictions through an API that applications query each time they start downloading a video chunk. An approach using an API-based architecture for cooperation between network operators and content providers has been recently proposed [9]. Collecting and analyzing various network information including user mobility, channel quality for millions of users and generating predictions in real time to be useful for video rate selection (and other possible applications) is a significant challenge. In order to scale, we envision that a practical system will have to partition the network into small areas and independently compute predictions for each area. However, this brings with it interesting protocol challenges in terms of localization and routing of requests to the appropriate area.

6. CONCLUSION

A cellular operator typically has access to a range of in-network measurements that can be potentially used for making short to mid-term bandwidth predictions. We show that when designing video adaptation algorithms, leveraging bandwidth predictions can significantly improve video QoE. We show that during startup, the proposed PBA algorithm promises to deliver more than 4x better video quality compared to heuristic-based algorithms. This is important since it has been shown that users typically abandon in the first few minutes of the video if the picture quality suffers. We also showed that a naive algorithm that solely uses prediction does not do well and that a combination of prediction with buffer occupancy and/or stability function is needed to extract the maximum benefit. Finally, we show that PBA algorithms with different stability functions can trade off stalls and stability while maintaining much improved average quality. For future work, we continue to address some of the open challenges in generating bandwidth predictions in real time and conveying them at scale.

7. REFERENCES

[1] Mobile throughput guidance signaling protocol. `https://tools.ietf.org/id/draft-flinck-mobile-throughput-guidance-00.txt`.

[2] A. Balachandran, V. Sekar, A. Akella, S. Seshan *et al.* Developing a predictive model of quality of experience for internet video. In *Proceedings of ACM SIGCOMM*, 2013.

[3] J. Chen, A. Ghosh, J. Magutt, and M. Chiang. QAVA: Quota Aware Video Adaptation. In *Proceedings of ACM CoNEXT*, 2012.

[4] J. Chen, R. Mahindra, M. A. Khojastepour, S. Rangarajan, and M. Chiang. A Scheduling Framework for Adaptive Video Delivery over Cellular Networks. In *Proceedings of ACM MobiCom*, 2013.

[5] Cisco. Cisco visual networking index: Forecast and methodology, 2013–2018, 2014. `http://tinyurl.com/mev32z8`.

[6] Citrix. Mobile analytics report, 2014. `http://tinyurl.com/n4qvgnf`.

[7] J. Erman, A. Gerber, K. K. Ramakrishnan, S. Sen, and O. Spatscheck. Over the Top Video: The Gorilla in Cellular Networks. In *Proceedings of ACM IMC*, 2011.

[8] T.-Y. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson. A Buffer-Based Approach to Rate Adaptation: Evidence from a Large Video Streaming Service. In *Proceedings of ACM SIGCOMM*, 2014.

[9] J. Jiang, X. Liu, V. Sekar, I. Stoica, and H. Zhang. EONA: Experience-oriented network architecture. In *Proceedings ACM HotNets*, 2014.

[10] J. Jiang, V. Sekar, and H. Zhang. Improving Fairness, Efficiency, and Stability in HTTP-based Adaptive Video Streaming with FESTIVE. In *Proceedings of ACM CoNEXT*, 2012.

[11] R. Margolies, A. Sridharan, V. Aggarwal, R. Jana *et al.* Exploiting mobility in proportional fair cellular scheduling: Measurements and algorithms. In *Proceedings of IEEE INFOCOM*, 2014.

[12] R. K. P. Mok, X. Luo, E. W. W. Chan, and R. K. C. Chang. QDASH: A QoE-aware DASH System. In *Proceedings of ACM MMSys*, 2012.

[13] A. J. Nicholson and B. D. Noble. Breadcrumbs: Forecasting mobile connectivity. In *Proceedings of ACM MobiCom*, 2008.

[14] Q. Xu, S. Mehrotra, Z. Mao, and J. Li. Proteus: Network performance forecast for real-time, interactive mobile applications. In *Proceeding of ACM MobiSys*, 2013.

[15] X. Yin, V. Sekar, and B. Sinopoli. Toward a principled framework to design dynamic adaptive streaming algorithms over http. In *Proceedings ACM HotNets*, 2014.