Adaptation logic for HTTP dynamic adaptive streaming using geo-predictive crowdsourcing for mobile users

Ran Dubin1 · Amit Dvir2 · Ofir Pele2,3 · Ofer Hadar1 · Itay Katz1 · Ori Mashiach1

1 Introduction

Dynamic Adaptive Streaming over HTTP (DASH) [1] is the HTTP Adaptive Streaming (HAS) standard. It has recently been adopted by YouTube (Google) and Netflix. DASH splits a video into chunks and encodes each into several quality representations. A client’s DASH application often has a smart Adaptation Logic (AL) module. The AL module is responsible for selecting the most suitable quality representation to enhance the client’s Quality of Experience (QoE) while considering factors such as the client’s buffer and playback delay. QoE is affected by factors such as the number of quality changes and their sizes. There is a tradeoff between increasing the video quality and buffering additional video segments. A client’s player often buffers a high number of segments to overcome network outages. Most current AL methods [2–10] estimate the next suitable segment based on estimates of previous segments without taking into account the future network characteristics. However, knowledge of geo-location network conditions can enable better decisions.

The term crowdsourcing was introduced by Howe [11] who defined crowdsourcing as the act of taking a task traditionally performed by a designated agent (such as an employee or a contractor) and outsourcing it by making an open call to an undefined but large group of people, especially from an online community. In the case of video adaptive streaming, crowdsourcing makes it possible to collect mobile network data anonymously and automatically. This is done using an application especially designed to improve the AL decision. Neidhardt et al. [12] reported that using many of the existing open datasets leads to low accuracy because of extreme outliers and few measurements for some of the cells. They noted that...
cellular location providers do not provide their complete data.

In this work, we present real-world, crowdsourcing datasets and test our proposed solution against state-of-the-art algorithms on them. Our work main goal is improving quality of experience while traveling without a known destination. We present six datasets, each of which contains samples of a single operator (T-Mobile, AT&T or Verizon) and from a single road (I100 or I405). All measurements were from Android cellphones of various users. The datasets were provided by WeFi LTD. The WeFi application collects granular information on mobile network performance and application usage from millions of devices, down to a 10 × 10 m geographical resolution. The datasets are public for academic users [13].

There are also disadvantages to relying on a crowdsourcing service. For example, a crowdsourcing service might not be reliable or trustworthy. This problem may be mitigated by relying on a trusted and reliable cloud service such as Amazon Web Services (AWS).

We propose a Geo-Predictive Adaptive Logic (GPAL) algorithm based on crowdsourcing data and show that it outperforms the state-of-the-art: Riiser [14], MASERATI [15], n-Predict [16], 1-Predict [16], PBA [17], MAL [10] and MaxBW [2] algorithms. It is worth noting that our crowdsourcing data were generated by users of a large scale, professional service and not by simulation.

The remainder of this paper is organized as follows: Sect. 2 summarizes related work. Section 3 presents our crowd algorithm. Section 4 details our dataset characteristics. Section 5 describes the experimental setup and results. Section 6 discusses future work and conclusions.

2 Related work

DASH AL is a well-investigated research topic. AL research can be roughly divided into two different groups: past estimate based AL and crowdsourcing based AL. Most work has investigated past estimate algorithms. Müller et al. [8] suggested a buffer based decision algorithm that uses the previous segment bandwidth estimates and the user’s current buffer duration to select a suitable quality representation for downloading. The Multicast Adaptation Logic (MAL) algorithm [10] uses a double Exponential Moving Average (EMA) algorithm. One

smooths the buffer size estimate and the other smooths the bandwidth estimate. This is done to select the most suitable segment. Although MAL was designed for multicast, it achieves good performance in unicast networks [10].

Crowdsourcing AL methods have attracted much less attention than past estimate based methods. Hung et al. [18] proposed a video streaming control mechanism based on location to overcome signal variations in train tunnels and underground areas. Geo-location frameworks that have the ability to predict future network conditions based on a bandwidth lookup service and similar concepts can be found in [16, 19–22]. Acharya et al. [23] evaluated rate-adaptation in a vehicular network based on signal strength and throughput at a location as an indicator of congestion. Curcio et al. [19] and Singh et al. [20] suggested server-side prediction algorithms for RTP streaming. Curcio et al. [19] suggested a framework with a predictive server that obtains route, speed, location and throughput from the client. However, this study was based on simulation rather than real-world data. Singh et al. [20] proposed building a Network Coverage Map Service (NCMS) to make rate-control decisions over a Real Time Protocol (RTP) using a server-side adaptation algorithm. Singh et al. however did not investigate performance on datasets with a higher geographical coverage or more diverse network connectivity conditions.

Yao et al. [21] showed that past bandwidth information is a good indicator of the actual bandwidth at a given location. Yao et al. found that location had greater influence than time, based on traces. Nevertheless, their performance evaluation did not take into account the number of switches or the playout buffer size. Furthermore, it was gathered from a small set of vehicles.

Riiser et al. [22] proposed constantly monitoring the receiver’s download rate and the associated GPS positional data to report a central database. This was argued to allow the user to better predict the near-future bandwidth availability and to create a video playout schedule based on the likely future availability bandwidth. The user bases his near future decision on the available bandwidth during the last interval to predict the available bandwidth during the next interval. The authors concluded that using past bandwidth lookups led to far fewer rebuffering events and stabler quality. Evensen et al. [24] suggested using the geographical location information to predict available bandwidth
accurately since the bandwidth is usually dependent on the user location. However, both systems [22, 24] often lead to inaccurate bandwidth prediction or pay little attention to user environments (e.g., time, humidity, speed). Therefore, Han et al. [15] investigated the extent to which the available user mobile channel bandwidth was affected by constraints including location, time, speed, humidity and cellular network type (3G/4G). Their scheme, called MASERATI, outperformed Pure-DASH and LoDASH, where Pure-DASH (equivalent to [22]) only used the download throughput and LoDASH (equivalent to [24]) used location based bandwidth predictions as in [22, 24].

Liu et al. [25] suggested comparing the segment fetch time with the media duration contained in the segment to detect congestion and probe the spare network capacity. Liu’s algorithm used conservative step-wise up switching and aggressive down switching. Hao et al. [16] suggested two algorithms: 1-predict and n-predict. The 1-predict algorithm uses the playout buffer and the next prediction to determine the most suitable representation to download. The n-predict algorithm uses the average throughput of the next n time steps as the algorithm’s current prediction. Hao et al. [16] evaluated Liu et al.’s algorithm and found that it achieved stable video quality but with a very low average bitrate. They showed that n-predict outperformed Liu et al’s algorithm as well as 1-predict. Zoë et al. [17] demonstrated that leveraging bandwidth predictions can significantly improve QoE. They designed an algorithm that combined bitrate prediction and rate stabilization. They showed that during startup, their algorithm had more than four times better video quality than heuristic-based algorithms.

Riiser et al. [14] recorded 3G mobile traces in Oslo, Norway, while traveling on different types of public transportation (metro, tram, train, bus and ferry). The traces, measured on the bus path between Ljøen and Oslo central station, Norway, had a total duration of about 220 minutes (almost 12500 samples). The available bandwidth was between 202bps and 6335kbps with an average of 2192kbps and STD of 1317kbps. This dataset has been used in several papers such as [30]. For our crowd sourcing algorithm we needed a bigger dataset both for the entire road and for every part of the road. Moreover, we had samples from different days of the week and various mobile devices. Therefore, we decided to only use our dataset. Note that, our DummyNet server (see Sect. 5.2) was equivalent to the Apache server in [14] so as to produce the same network conditions for all tests. Table 1 summarizes the papers presented above.

3 The geo-predictive adaptive logic (GPAL) algorithm

We define the user playout buffer as \( B(t) \). The goal of the AL modules is to maximize the overall quality of the stream, while eliminating rebuffering \( (B(t) > 0) \). We measure the quality in terms of its eMOS score [31, 32] as can be seen in Eq. 1.

\[
eMOS = \max(0.81\mu - 0.96\sigma - 4.95\phi + 0.17, 0)
\]

\[
\phi = \frac{7 \max(\frac{\ln(F_{\text{freq}})}{6} + 1, 0) + \min(F_{\text{avg}}, 15)/15}{8}
\]

The eMOS combines the re-buffering effect on the users experience with the influence of the selected quality representation (for more information see [31]). The eMOS range is between 0 and 5.84 (the highest quality). In these equations \( F_{\text{freq}} \) and \( F_{\text{avg}} \) represent the number of re-buffering events relative to the number of segments and the average duration over all re-buffering, respectively. The quality effect on the user is expressed as \( (\mu) \) the normalized average quality level and the \( (\sigma) \) the standard deviation of the normalized quality level. Note that eMOS [31, 32] is an estimated MOS based on several video parameters such as quality and re-buffering. In our model, we used an estimated Mean Opinion Score (eMOS) formula (Eq. 1) based on the work of Claey et al. [31].

Finally, our algorithm tries to maximize the eMOS subject to constrains as shown in Eq. 2.

\[
\max(\text{eMOS}) s.t. \quad \forall t > t_{\text{start}} \quad 0 < B(t) \leq B_{\text{max}}
\]

The GPAL algorithm, Algorithm 1, determines the representation of the next media segment to be fetched. The algorithm estimates the current segment download path based on the client’s location and speed. It predicts the future path network bandwidth conditions based on the client’s playout buffer and the crowd estimated bandwidth. The algorithm calculates the playout buffer fullness ratio \( (B_p) \) based on the maximum between the current buffer levels divided by the maximum buffer size allocation. The bandwidth coefficients (thresholds) in our algorithm are taken from MaxBW [2].
4 Dataset

The WeFi datasets contain samples from the California I110 and I405 interstates. The I110 is an interstate highway in the Los Angeles area and connects San Pedro and the port of Los Angeles with downtown Los Angeles and Pasadena. The I405 is a major north-south interstate highway in Southern California. Table 2 summarizes the number of samples on each road over a operator.

Each sample of the dataset contains longitude, latitude, data throughput and data size. A large number of different applications generated the data. Most of the applications either regularly send low rate updates or are in the idle state (sending keep-alive messages). Thus, most of the samples’ data throughput are significantly lower than the real channel throughput. Thus, for each road interval $x$ we estimate its channel throughput, $E_x$, using Eq. 3.

$$E_x = \frac{\sum_{s \in x} D_s \cdot A_s}{\sum_{s \in x} D_s}$$  \hspace{1cm} (3)$$

where $D_s$ is sample $s$’ data size and $A_s$ is sample $s$’ data throughput. This estimate gives more weight to high data throughput as there are more data transmitted in high data throughput. It is noteworthy that practically all samples of long-duration TCP connections are the ones that are used almost exclusively for the estimation of the absolute throughput. WeFi used our equation and verified experimentally that the estimate is accurate. Figures 1, 2 depict data throughputs vs. estimated channel throughputs along the roads (for 12 m segments). We can see that the estimated channel throughput varies along the roads and are high enough for video streaming in different video quality representations.

The dataset was split into the different operators. Then, each of these datasets was split into a train set and a test set. The train set contained samples from Wednesday, Thursday, Friday and Saturday. The test set contained samples from Sunday, Monday and Tuesday. The train sets were used as the crowd source data and the test sets were used for bandwidth generation using the testing phase via Dummynet [33].

4.1 Interstate I110

The interstate heat map is illustrated in Fig. 3 which shows that the road throughput can vary between 0.5 $-$ 5 [Mb/s]. Figure 3 depicts the measured bandwidth of the path (average and STD). We define this bandwidth path as I110.

The median throughput of the interstate is 0.86 [Mb/s], the average throughput is 1.585 [Mb/s] and the STD is 2. That is, the path has many fluctuations. Thus, it is challenging for adaptive streaming clients to adapt to its network conditions.

Figure 3 depicts the throughput density and the sample densities along the route. We split the throughput density into fixed bins from 0 to the maximum observed
| Paper                        | Streaming protocol                      | Idea                                                                 | Trigger                                      | Action                          | Quality adjustment                | Compared algorithms | Observed metrics                                    |
|-----------------------------|-----------------------------------------|----------------------------------------------------------------------|----------------------------------------------|----------------------------------|-----------------------------------|---------------------|-----------------------------------------------------|
| Singh et al. [20] - Geo-   | RTP/UDP + Temporal Maximum Media Stream | Avoiding buffer underrun—client looks ahead at locations in its vicinity for bad coverage | Future coverage hole                        | Client Pre-Buffer                | Client media rate switch according to available throughput in the coverage hole | No adaptation (RTCP), rate switching GLASS, late scheduling GLASS, Omniscient (optimal) | packet loss rate, average receive rate, Y component of the PSNR, throughput |
| location Assisted Streaming System (GLASS) Rate-Switching                        | (TMMBR)                                 |                                                                     |                                              |                                  |                                    |                     |                                                     |
| Yao et al. [21] - BW-MAP-TFRC | Adaptive TCP streaming with TCP Friendly Rate Control (TFRC) [26] | Avoiding packet loss—client updates the server when it changes its location. The server determines the average bandwidth at that location in the past | Location changed by client followed by a new BW value | Server changes its sending rate | Short freezing of the TFRC and disabling the normal operation of TFRC when needed | TFRC and BW-MAP-TFRC |                                                     |
| Rüser et al. [22]          | Apple Live HTTP                          | Minimizing rapid fluctuations in quality and avoiding buffer underrun—client’s estimate of the number of bytes that it can download during the remaining time of the trip | Client receives a sequence of bandwidth averages for its whole path | Client plans which quality levels to use | Apple Live HTTP mechanism | Buffer-based reactive, history-based prediction, omniscient prediction (optimal) | Buffer size, selected representations               |
| Han et al. [15] - MASERATI | DASH                                    | Avoiding frequent or large video quality changes                     | The algorithm finds the most similar database entry and estimates the available bandwidth | The bit rate of the next video segment is defined by that bandwidth | DASH Adaptation mechanism         | Pure-DASH, location-based DASH (LoDASH) [22], MASERATI | Playout Success Rate, Quality of Segments, Frequency of Quality Changes, Degree of Changed Quality Level |
| Hao et al. [16] - 1-predict, n-predict | DASH                                    | Achieving continuous playback—DASH Based algorithm with an additional function to anticipate future path and bandwidth, and to determine the predicted rate | The server calculates the possible bandwidth and sends it to the client | DASH Client applies the best quality level it can afford | DASH Adaptation mechanism         | Liu et al. [25], Adobe Open Source Media Framework (OSMF), 1-Predict, N-Predict | Segment representation Level, Ratio of bandwidth utilization, rate of video quality level shift |
throughput, 10 [Mb/s]. It is clear that lower throughput in the ranges of 0 – 2 [Mb/s] are more likely while throughput above 5 [Mb/s] are less common. Figure 3 shows the sample densities along the route. From 23 km the sample densities decrease.

### 4.2 Interstate I405

The I405 interstate is shorter but has a higher number of samples than the I110 interstate (see Table 2). The interstate heat map is illustrated in Fig. 4 which shows that the road throughput varies between 0.5 – 5 [Mb/s]. Figure 4 depicts the measured bandwidth of the path (average and STD). We define this bandwidth path as I405A.

The median throughput of the interstate is 1.97[Mb/s], the average throughput is 2.63 [Mb/s] and the STD is 2.15. The I405 interstate has a higher throughput average than I110. The STD is slightly higher.
Adaptation logic for HTTP

Figure 4 illustrates the throughput density and the sample densities along the route. We split the throughput density into fixed bins from 0 to the maximum observed throughput 10 [Mb/s]. The table shows that the I405 throughput density is different from the I110 throughput density and the throughput is better spread between 0.5 – 2.5 [Mb/s]. Figure 4 depicts the density of the samples along the route.

5 Experiments and results

We describe our experimental setup and video representation information in Sect. 5.1. We discuss our experimental results in Sect. 5.2.

5.1 Experimental setup

This section describes our experimental settings and video encoding configuration. We used the Big Buck Bunny (BBB) [34] video encoded into fixed duration segments of 2 s. Table 3 illustrates the BBB available representation stored in the streaming server. The client playout buffer duration was set to 30 s.

Figure 5 illustrates our experimental setup. First, the user requests (using VLC [35]) the video MPD file from the HTTP server via the Dummynet [33] which shapes the traffic. The Dummynet network scenario does not use any information from the train dataset (Wednesday, Thursday, Friday and Saturday). The Dummynet network scenario is based on the test dataset (Sunday, Monday and Tuesday).

After the client receives it, the adaptation logic algorithm requests the crowd estimate from the PostgreSQL geo-predictive server (taken from the train set). Then, the user sends a request to the server using a simple API implementation which only sends the following information to the server: the search radius (250 m), the user’s current location and the estimated end point (which depends on the user’s average speed). The geo-predictive module predicts the average throughput. Since this API is very lightweight, the process delay is negligible.
We do not assume we know the route. Therefore, we used a batch fetching mechanism. That is, before the current segment download ends, we fetch the crowd estimate for the next segment. Each adaptation logic can analyze the data or use the API differently but the fetching optimization is beyond the scope of this study. As a result, the segment download is delayed according to the network conditions.

To compare our work to state-of-the-art algorithms we used the same segment fetching schema as these works, where the client downloads each segment one after the other.
5.2 Experimental Results

In Eq. 2 we stated our goal. In our experiments $B_{\text{max}}$ was 30 s. All the algorithms implemented the constraints in Eq. 2.

Figure 6 depicts the eMOS score (Eq. 1) for all the algorithms. Our algorithm, GPAL, has the best performance for all datasets except I110 over T-Mobile, where it was slightly outperformed by MAL. MAL and Riiser also exhibited good performance. We can see several cases where the eMOS score is zero. This is especially noticeable on I110 road over AT&T and Verizon operators where the cellular network conditions were hard.

Our main goal was to improve the quality of experience while traveling without a known destination. Additionally, we tested the performance of a prefetch mechanism. We added planning inspired by the ideas of [22]. Based on the user’s speed, we estimated the user’s location on the road of the next segment request. The adaptation logic algorithm decides the qualities both for the current segment and the next segment and requests the average quality for the current location. We tested this method using both our GPAL algorithm and the Riiser algorithm. The results are presented in Fig. 7. Prefetching did not improve eMOS results in most cases and even lowered it in some cases.

Tables 4, 5, 6, 7, 8 and 9 present the detailed numerical results of our experiments. For each evaluation metric, the best algorithm’s results (highest for average bitrate, eMOS, PSNR, $\mu$ and lowest for all the others) are marked in bold. We can see that in many cases algorithms had very good results in one metric at the cost of having very low results in another metric, thus achieving an overall low quality of experience. For example, 1-Predict, PBA and MaxBw PSNR and average bitrate results were excellent. However, they had a high number of rebuffering events and relatively high number of switches which reduced the eMOS scores to low values, especially on the I110 road. Maserati and MaxBW had a high number of switches and their other QoE parameters were not especially good and thus the
Fig. 7 Comparison of GPAL and Riiser algorithms with and without prefetching mechanism. Prefetching did not improve eMOS results in most cases and even lowered it in some cases.

### Table 4 T-Mobile user results, over road I110

| Algorithm  | Average bitrate (Kbps) | eMOS | Average PSNR | Num of buffering | Num of switch up | Num of switch down | Sum: up + down | μ   | σ   | φ   |
|------------|------------------------|------|---------------|------------------|-----------------|-------------------|----------------|-----|-----|-----|
| GPAL       | 236.55                 | 0.73 | 33.26         | 4.00             | 32.00           | 32.00             | 64.00          | 3.64| 0.89| 0.30|
| Riiser     | 125.56                 | 0.00 | 28.89         | 4.00             | 10.00           | 9.00              | 19.00          | 2.22| 1.01| 0.31|
| Maserati   | 238.05                 | 0.26 | 32.66         | 4.00             | 42.00           | 47.00             | 89.00          | 3.45| 1.23| 0.31|
| N-Predict  | 95.41                  | 0.00 | 27.04         | 4.00             | 6.00            | 6.00              | 12.00          | 1.73| 1.01| 0.31|
| 1-Predict  | **624.22**             | 0.00 | 36.70         | 20.00            | 49.00           | 31.00             | 80.00          | 5.04| 2.25| 0.54|
| PBA        | 576.11                 | 0.28 | **37.58**     | 20.00            | 38.00           | 24.00             | 62.00          | **5.26**| 1.52| 0.54|
| MAL        | 407.71                 | **0.88** | 35.47     | **3.00**         | **9.00**        | **2.00**          | **11.00**      | 4.35| 1.55| 0.27|
| MaxBw      | 604.04                 | 0.00 | 36.71         | 12.00            | 69.00           | 67.00             | 136.00         | 5.07| 2.10| 0.47|

### Table 5 AT&T user results, over road I110

| Algorithm  | Average bitrate (Kbps) | eMOS | Average PSNR | Num of buffering | Num of switch up | Num of switch down | Sum: up + down | μ   | σ   | φ   |
|------------|------------------------|------|---------------|------------------|-----------------|-------------------|----------------|-----|-----|-----|
| GPAL       | 243.98                 | **0.73** | 33.35         | **4.00**         | 25.00           | 22.00             | 47.00          | 3.70| **0.95**| **0.31**|
| Riiser     | 150.24                 | 0.17 | 30.50         | **4.00**         | 16.00           | 10.00             | 26.00          | 2.88| 1.13| 0.41|
| Maserati   | 172.32                 | 0.00 | 29.28         | 12.00            | 33.00           | 37.00             | 70.00          | 2.44| 1.6 | 0.47|
| N-Predict  | 150.91                 | 0.00 | 28.86         | **4.00**         | **3.00**        | **5.00**          | **8.00**       | 3.33| 1.52| 0.31|
| 1-Predict  | 385.92                 | 0.00 | 32.48         | 28.00            | 45.00           | 25.00             | 70.00          | 3.62| 2.4 | 0.59|
| PBA        | **424.91**             | 0.00 | **34.42**     | 28.00            | 34.00           | 23.00             | 57.00          | **4.15**| 2.04| 0.60|
| MAL        | 308.73                 | 0.00 | 33.24         | 16.00            | 20.00           | 15.00             | 35.00          | 3.70| 1.68| 0.51|
| MaxBw      | 404.14                 | 0.00 | 32.53         | 8.00             | 60.00           | 53.00             | 113.00         | 3.65| 2.49| 0.41|
Table 6  Verizon user results, over road I110

| Algorithm | Average bitrate (Kbps) | eMOS | Average PSNR | Num of buffering | Num of switch up | Num of switch down | Sum: up + down µ | σ | φ |
|-----------|------------------------|------|---------------|------------------|-----------------|------------------|------------------|------|------|
| GPAL      | 216.77                 | 1.90 | 32.47         | 0.00             | 51.00           | 47.00            | 98.00            | 3.37 | 1.04 |
| Riiser    | 191.93                 | 0.00 | 30.10         | 95.00            | 18.00           | 13.00            | 31.00            | 2.51 | 1.23 |
| Maserati  | 285.49                 | 0.00 | 31.81         | 12.00            | 47.00           | 60.00            | 107.00           | 3.27 | 1.47 |
| N-Predict | 140.40                 | 0.62 | 28.57         | 0.00             | 7.00            | 11.00            | 18.00            | 2.19 | 1.38 |
| 1-Predict | 336.25                 | 0.00 | 32.91         | 32.00            | 40.00           | 31.00            | 71.00            | 1.97 | 0.61 |
| PBA       | 388.29                 | 0.00 | 32.61         | 152.00           | 40.00           | 24.00            | 64.00            | 3.62 | 1.21 |
| MAL       | 238.07                 | 0.00 | 30.94         | 14.00            | 27.00           | 13.00            | 40.00            | 2.93 | 1.73 |
| MaxBw     | 231.58                 | 0.00 | 30.93         | 12.00            | 52.00           | 50.00            | 102.00           | 2.97 | 1.75 |

Table 7  T-Mobile user results, over road I405

| Algorithm | Average bitrate (Kbps) | eMOS | Average PSNR | Num of buffering | Num of switch up | Num of switch down | Sum: up + down µ | σ | φ |
|-----------|------------------------|------|---------------|------------------|-----------------|------------------|------------------|------|------|
| GPAL      | 1041.39                | 4.87 | 41.40         | 0.00             | 33.00           | 30.00            | 63.00            | 7.0  | 1.01 |
| Riiser    | 807.20                 | 3.83 | 39.70         | 0.00             | 16.00           | 10.00            | 26.00            | 6.21 | 1.43 |
| Maserati  | 423.36                 | 2.04 | 35.12         | 0.00             | 42.00           | 53.00            | 95.00            | 4.38 | 1.75 |
| N-Predict | 233.63                 | 1.56 | 32.30         | 0.00             | 9.00            | 6.00             | 15.00            | 3.28 | 1.32 |
| 1-Predict | 1171.55                | 1.12 | 39.98         | 8.00             | 49.00           | 34.00            | 83.00            | 6.80 | 2.63 |
| PBA       | 1113.43                | 0.90 | 40.66         | 24.00            | 36.00           | 25.00            | 61.00            | 6.80 | 2.04 |
| MAL       | 887.93                 | 3.37 | 39.48         | 0.00             | 14.00           | 7.00             | 21.00            | 6.23 | 1.93 |
| MaxBw     | 1115.30                | 1.76 | 40.08         | 4.00             | 66.00           | 61.00            | 127.00           | 6.70 | 2.40 |

Table 8  AT&T user results, over road I405

| Algorithm | Average bitrate (Kbps) | eMOS | Average PSNR | Num of buffering | Num of switch up | Num of switch down | Sum: up + down µ | σ | φ |
|-----------|------------------------|------|---------------|------------------|-----------------|------------------|------------------|------|------|
| GPAL      | 950.13                 | 4.59 | 40.94         | 0.00             | 26.00           | 22.00            | 48.00            | 6.72 | 1.06 |
| Riiser    | 693.47                 | 2.89 | 38.20         | 0.00             | 17.00           | 12.00            | 29.00            | 5.58 | 1.88 |
| Maserati  | 416.03                 | 2.07 | 35.17         | 0.00             | 45.00           | 52.00            | 97.00            | 4.36 | 1.69 |
| N-Predict | 190.91                 | 1.12 | 30.76         | 0.00             | 6.00            | 6.00             | 12.00            | 2.96 | 1.50 |
| 1-Predict | 1010.36                | 0.50 | 39.12         | 11.00            | 48.00           | 29.00            | 77.00            | 6.29 | 2.61 |
| PBA       | 1231.16                | 0.02 | 40.31         | 44.00            | 40.00           | 25.00            | 65.00            | 6.88 | 2.11 |
| MAL       | 790.96                 | 3.44 | 39.14         | 0.00             | 13.00           | 7.00             | 20.00            | 6.03 | 1.68 |
| MaxBw     | 1013.60                | 0.20 | 39.05         | 16.00            | 59.00           | 52.00            | 111.00           | 6.28 | 2.63 |

Table 9  Verizon user results, over road I405

| Algorithm | Average bitrate (Kbps) | eMOS | Average PSNR | Num of buffering | Num of switch up | Num of switch down | Sum: up + down µ | σ | φ |
|-----------|------------------------|------|---------------|------------------|-----------------|------------------|------------------|------|------|
| GPAL      | 1240.38                | 5.24 | 42.23         | 0.00             | 36.00           | 29.00            | 65.00            | 7.47 | 1.02 |
| Riiser    | 853.99                 | 3.71 | 39.79         | 0.00             | 15.00           | 10.00            | 25.00            | 6.27 | 1.60 |
| Maserati  | 521.64                 | 2.28 | 36.25         | 0.00             | 59.00           | 68.00            | 127.00           | 4.81 | 1.85 |
| N-Predict | 324.58                 | 1.64 | 33.60         | 0.00             | 13.00           | 10.00            | 23.00            | 3.79 | 1.67 |
| 1-Predict | 1504.06                | 2.84 | 42.09         | 4.00             | 38.00           | 24.00            | 62.00            | 7.85 | 2.25 |
| PBA       | 1498.48                | 3.09 | 43.08         | 16.00            | 34.00           | 26.00            | 60.00            | 8.04 | 1.10 |
| MAL       | 1124.72                | 4.40 | 41.25         | 0.00             | 8.00            | 0.00             | 8.00             | 7.05 | 1.53 |
| MaxBw     | 1490.58                | 4.54 | 42.26         | 0.00             | 60.00           | 52.00            | 112.00           | 7.87 | 2.10 |
overall eMOS scores were low. Riiser and MAL algorithms had good performance, where the QoE parameters were relatively balanced. Finally, GPAL had the least number of buffering events (except on I110 over T-Mobile where it was 2nd best). Additionally, GPAL had balanced all other QoE parameters and thus its QoE was the best.

6 Conclusion

We proposed a new crowd-based algorithm called GPAL. We evaluated our algorithm and state-of-the-art algorithms on large, real-life, crowdsourcing datasets. Our algorithm, GPAL, outperformed all other state-of-the-art algorithms. Thus an optimal adaptation logic should estimate the distance between the current conditions and the cloud conditions. Our future work will aim at designing an adaptation algorithm that can leverage the advantages of past download algorithms with crowd knowledge based on the conclusions drawn from this work. An interesting approach would be to implement machine learning algorithms (similar to Claeyss et al. [31]) combined with crowd data. Another possible direction would be to harness a client-side pre-fetch and HTTP2 server-side push mechanism based on crowd knowledge similar to [36–41].

References

1. ISO/IEC: Information technology—dynamic adaptive streaming over HTTP (DASH) (2014)
2. Müller, C., Lederer, S., Timmerer, C.: An evaluation of dynamic adaptive streaming over http in vehicular environments. In: Proceedings of the 4th Workshop on Mobile Video, pp. 37–42 (2012)
3. Mok, R.K.P., Luo, X., Chan, E.W.W., Chang, R.K.C.: Qdash: a qoe-aware dash system. In: Proceedings of the 3rd Multimedia Systems Conference, ACM, pp. 11–22 (2012)
4. Dubin, R., Hadar, O., Dvir, A.: The effect of client buffer and mbr consideration on dash adaptation logic. In: WCNC, pp. 2178–2183 (2013)
5. Wu, T.T., Leeuwijk, W.: Factor selection for reinforcement learning in http adaptive streaming. In: MultiMedia modeling, vol. 8325 of Lecture Notes in Computer Science, pp. 553–567. Springer, Berlin (2014)
6. Mueller, C., Lederer, S., Timmerer, C.: A proxy effect analysis and fair adaptation algorithm for multiple competing dynamic adaptive streaming over http clients. In: Proceedings of the IEEE Conference on Visual Communications and Image Processing Conference (VCIP 2012); IEEE, Piscataway (NJ), pp. 6 (2012)
7. De Cicco, L., Caldararo, V., Palmisano, V., Mascolo, S.: Elastic: a client-side controller for dynamic adaptive streaming over http (dash). In: Packet Video Workshop (PV), 20th International, pp. 1–8. IEEE (2013)
8. Müller, C., Lederer, S., Timmerer, C.: An evaluation of dynamic adaptive streaming over http in vehicular environments. In: Proceedings of the 4th Workshop on Mobile Video, pp. 37–42. ACM (2012)
9. Dubin, R., Hadar, O., Ben-Moshe, B., Dvir, A.: A novel adaptive logic for dynamic adaptive streaming over http network. In: IEEEI (2014)
10. Dubin, R., Dvir, A., Hadar, O., Harel, N., Barkan, R.: Multicast adaptive logic for dynamic adaptive streaming over http network. In: Workshop Communication and Networking Techniques for Contemporary Video (INFOCOM workshop) (2015)
11. Howe, J.: Crowdsourcing: why the power of the crowd is driving the future of business. J. Consum. Mark. 26(4), 305–306 (2009)
12. Neidhardt, E., Uzun, A., Bareth, U., Kupper, A.: Estimating locations and coverage areas of mobile network cells based on crowdsourced data. In: Wireless and mobile networking conference (WMNC), 2013 6th Joint IFIP, IEEE, pp. 1–8 (2013)
13. WeFi LTD. WeFi data set. http://www.ariel.ac.il/sites/amitd/data-sets
14. Riiser, H., Vigmestad, P., Grewitz, C., Halvorsen, P.: Commute path bandwidth traces from 3G networks: analysis and applications. In: Proceedings of the 4th ACM Multimedia Systems Conference, pp. 114–118. ACM (2013)
15. Han, D., Han, J., Im, Y., Kwak, M., Kwon, T.T., Choi, Y.: Maserati: mobile adaptive streaming based on environmental and contextual information. In: Proceedings of the 8th ACM international workshop on Wireless network testbeds, experimental evaluation & characterization, pp. 33–40. ACM (2013)
16. Hao, J., Zimmermann, R., Ma, H.: Gube: geo-predictive video streaming over http in mobile environments. In: Proceedings of the 5th ACM Multimedia Systems Conference, pp. 259–270. ACM (2014)
17. Zou, X.K., Erman, J., Gopalakrishnan, V., Halepovic, E., Jana, R., Jin, X., Refined, J., Sinha, R.K.: Can accurate predictions improve video streaming in cellular networks? In: Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications, HotMobile ‘15, pp. 57–62 (2015)
18. Huang, C.M., Wu, S.W., Yu, Y.T.: Lbvs-t: A location-based video streaming control scheme for trains. In: IEEE 10th International Conference on Wireless and mobile computing, networking and communications (WiMob), pp. 679–684 (2014)
19. Curcioa, I.D.D., Vadakital, V.K.M., Hannukela, M.M.: Geo-predictive real-time media delivery in mobile environment. In: Proceedings of the 3rd workshop on mobile video delivery, ACM, pp. 3–8 (2010)
20. Singh, V., Ott, J., Curcio, I.D.D.: Predictive buffering for streaming video in 3G networks. In: World of wireless, mobile and multimedia networks (WoWMoM), 2012 IEEE International Symposium on a, pp. 1–10 (2012)
21. Yao, J., Kanhere, S.S., Hassan, M.: Improving qos in high-speed mobility using bandwidth maps. IEEE Trans. Mob. Comput. 11(4), 603–617 (2012)
22. Riiser, H., Endestad, T., Vigmestad, P., Grewitz, C., Halvorsen, P.: Video streaming using a location-based bandwidth-lookup service for bitrate planning. ACM Trans. Multimed. Comput. Commun. Appl. (TOMCCAP) 8(3), 24 (2012)
23. Acharya, P.A.K., Sharma, A., Belding, E.M., Almroth, K.C., Papagiannaki, K.: Congestion-aware rate adaptation in wireless networks: a measurement-driven approach. In: 5th Annual IEEE Communications Society Conference on sensor, mesh and ad hoc communications and networks. SECON’08, pp. 1–9 (2008)
24. Evensen, K., Petlund, A., Riiser, H., Vigmestad, P., Kaspar, D., Grewitz, C., Halvorsen, P.: Mobile video streaming using location-based network prediction and transparent handover. In: International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV), pp. 21–26 (2011)
25. Liu, C., Bouazizi, I., Gabbouj, M.: Rate adaptation for adaptive http streaming. In: ACM multimedia systems, pp. 169–174, CA, USA (2011)
Adaptation logic for HTTP

26. Handley, M., Floyd, S., Padhye, J., Widmer, J.: Tcp friendly rate control (tfrc): protocol specification. Technical report (2002)
27. ITUT SG12. The e-model, a computational model for use in transmission planning. Recommendation G 107 (2005)
28. Jiang, J., Sekar, V., Zhang, H.: Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive. In: Proceedings of the 8th international conference on Emerging networking experiments and technologies, ACM, pp. 97–108 (2012)
29. Huang, T.Y., Johari, R., McKeown, N., Trunnell, M., Watson, M.: A buffer-based approach to rate adaptation: evidence from a large video streaming service. In: Proceedings of the 2014 ACM conference on SIGCOMM, ACM, pp. 187–198 (2014)
30. Estevez, A.G., Carlsson, N.: Geo-location-aware emulations for performance evaluation of mobile applications. In: 11th Annual Conference on wireless on-demand network systems and services (WONS), pp. 73–76 (2014)
31. Claeys, M., Latré, S., Famaey, J., Wu, T., Van Leekwijck, W., De Turck, F.: Design and optimisation of a (fa) q-learning-based http adaptive streaming client. Connect. Sci. 26(1), 25–43 (2014)
32. Claeys, M., Latré, S., Famaey, J., De Turck, F.: Design and evaluation of a self-learning HTTP adaptive video streaming client. IEEE Commun. Lett. 18(4), 716–719 (2014)
33. Carbone, M., Rizzo, L.: Dummynet. http://info.iet.unipi.it/~luigi/dummynet (2010)
34. Big Buck Bunny video. http://www.bigbuckbunny.org (2008)
35. Müller, C., Timmerer, C.: A vlc media player plugin enabling dynamic adaptive streaming over http. In: ACM Multimedia, pp 723–726, Arizona, USA (2011)
36. Dandapat, S.K., Jain, S., Ganguly, N., Choudhury, R.R.: Distributed content storage for just-in-time streaming. In: SIGCOMM—Poster, pp. 77–78 (2012)
37. Dandapat, S.K., Pradhan, S., Ganguly, N., Choudhury, R.R.: Sprinkler: distributed content storage for just-in-time streaming. In: Proceeding of the 2013 Workshop on cellular networks: operations, challenges, and future design, CellNet ’13, ACM, pp. 19–24, New York, NY, USA (2013)
38. Mahmood, A., Casetti, C., Chiasserini, C.F., Giaccone, P., Härr, J.: Mobility-aware edge caching for connected cars. In: WONS 2016, 12th IEEE Conference on Wireless On-Demand Networks systems and Services, 20–22 January 2016, Cortina d’Ampezzo, Italy, Cortina d’Ampezzo, ITALIE, 01 (2016)
39. Arai, S., Fadlullah, Z.M., Ngo, T., Nishiyama, H., Kato, N.: An efficient method for minimizing energy consumption of user equipment in storage-embedded heterogeneous networks. Wireless Communications, IEEE 21(4), 70–76 (2014)
40. Zhang, F., Xu, C., Zhang, Y., Ramakrishnan, K.K., Mukherjee, S., Yates, R., Nguyen, T.: Edgebuffer: caching and prefetching content at the edge in the mobilityfirst future internet architecture. In: World of wireless, mobile and multimedia networks (WoWMoM), 2015 IEEE 16th International Symposium on a, pp. 1–9 (2015)
41. Guan, Y., Xiao, Y., Feng, H., Shen, C.-C., Cimini Jr., L.J.: Mobicacher: mobility-aware content caching in small-cell networks (2014)