A Case for a Coordinated Internet Video Control Plane

Xi Liu, Florin Dobrian, Henry Milner
Conviva

Junchen Jiang
CMU

Vyas Sekar
Intel Labs

Ion Stoica
Conviva
UC Berkeley

Hui Zhang
Conviva
CMU

ABSTRACT

Video traffic already represents a significant fraction of today's traffic and is projected to exceed 90% in the next five years. In parallel, user expectations for a high quality viewing experience (e.g., low startup delays, low buffering, and high bitrates) are continuously increasing. Unlike traditional workloads that either require low latency (e.g., short web transfers) or high average throughput (e.g., large file transfers), a high quality video viewing experience requires *sustained* performance over *extended* periods of time (e.g., tens of minutes). This imposes fundamentally different demands on content delivery infrastructures than those envisioned for traditional traffic patterns. Our large-scale measurements over 200 million video sessions show that today's delivery infrastructure fails to meet these requirements: more than 20% of sessions have a rebuffering ratio $\geq 10\%$ and more than 14% of sessions have a video startup delay ≥ 10s. Using measurement-driven insights, we make a case for a video control plane that can use a global view of client and network conditions to dynamically optimize the video delivery in order to provide a high quality viewing experience despite an unreliable delivery infrastructure. Our analysis shows that such a control plane can potentially improve the rebuffering ratio by up to $2\times$ in the average case and by more than one order of magnitude under stress.

Categories and Subject Descriptors

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

General Terms

Design, Performance, Measurement

Keywords

Video, CDNs, Control plane

1. INTRODUCTION

Over the last few years, video traffic has quickly become the dominant fraction of Internet data traffic. Studies show that Netflix alone accounts for more than 20% of the US Internet traffic [42] and projections show that by 2014, video traffic will constitute more than 90% of the total traffic on Internet [2]. These estimates are for *streaming* traffic (including both live and video-on-demand services) and does not include offline video downloads (e.g., via shared upload or P2P services).

User expectations in streaming video workloads impose fundamentally different service requirements on the network infrastructure compared to traditional data traffic. Traditional workloads focus either on latency (e.g., interactive sessions or short web transfers) or on transfer completion time (e.g., long file transfers). In contrast, latency is less critical in streaming video because application data units are large enough to amortize latency effects. Similarly, overall completion time does not really reflect the actual user experience as it does not capture rebuffering-induced interruptions; we know that users are sensitive to buffering as a 1% increase in buffering can lead to more than a 3 minutes reduction in expected viewing time [21]. Streaming video not only introduces new quality metrics at the network- and user-level, but also requires that this quality (e.g., high bitrates with low rebuffering) be *sustained* over extended periods of time as typical videos span multiple minutes.

In addition to the improvements last-mile connectivity, a key driver for the rapid explosion of streaming video traffic has been the shift from specialized streaming protocols and infrastructure (e.g., Windows Media Services, RealNetworks, RTMP) to HTTP chunk-based streaming protocols (e.g., [5, 42]). This use of a commodity service dramatically decreases the cost of dissemination and the barrier of entry by allowing content providers to leverage existing HTTP CDN infrastructures to deliver content to a wide audience. Furthermore, the reliance on HTTP also implies the ability to support multiple viewing platforms as HTTP support is ubiquitous.

Unfortunately, there is a mismatch between the requirements of video streaming and the architecture of today's HTTP-based video delivery infrastructures, both at the ISP and CDN level. Using fine-grained client-side measurements from over 200 million client viewing sessions, we find that 20% of these sessions experience a rebuffering ratio of $\geq 10\%$, 14% of users have to wait more than 10 seconds for video to start up, more than 28% of sessions have an average bitrate less than 500Kbps, and 10% of users fail to see any video at all.

Analyzing the causes of these performance problems reveals:

- significant spatial diversity in CDN performance and availability across different geographical regions and ISPs,
- substantial temporal variability in the CDN performance and client-side network performance, and
- poor system response to overload scenarios when there are "hotspots" of client arrivals in particular regions or ISPs.

Our overarching goal is to meet the demands for a high-quality viewing experience despite an unreliable video delivery infrastructure. In this context, the design space for optimizing video delivery quality consists of three high-level dimensions:

1. *What* parameters can we adapt; e.g., bitrate, CDN?
2. *When* are these parameters optimized; e.g., at video startup or midstream?

SIGCOMM'12, August 13–17, 2012, Helsinki, Finland.

3. *Who* chooses these parameters; e.g., client or server?

The above observations regarding CDN variability across space and time suggest that purely server- or client-driven selection and adaptation are unlikely to be sufficient. To this end, we envision a video control plane that can use a *global* view of network and CDN performance to dynamically assign clients a suitable choice of CDN and bitrate that optimizes the video delivery. Beyond the performance benefits, such a control plane also offers content providers more flexibility in instrumenting fine-grained policies; e.g., providing higher quality service to premium customers under load, ensuring that certain types of content is only accessible with specific geographical regions, or taking into account the cost-performance tradeoffs that different CDNs have to offer [29].

Realizing such a control plane is challenging, and thus a natural first question is whether this exercise is worthwhile. To this end, we use a measurement-driven framework to extrapolate the potential for improvement in video quality. We observe that there is significant potential and that even just choosing a CDN more optimally can reduce the average rebuffering ratio by $2\times$ in the common case and more than $10\times$ under extreme scenarios.

We would also like to confirm that these gains are not merely hypothetical. To do so, however, we need to concretely specify aspects of the control plane such as the allocation algorithms, performance estimators, and policy functions. To this end, we present one specific realization of such a control plane to illustrate the benefits. Our choices in this respect are far from ideal and have to necessarily embed several simplifying assumptions. We believe this exercise is still valuable because our goal is to make a *case for* a control plane, rather than present a reference design and implementation. Our simulations confirm that such an approach can outperform other options in the design space for optimizing video delivery in both common and extreme load scenarios.

Contributions and Roadmap: To summarize, our key contributions are:

- Measurements to expose the shortcomings of today's video delivery infrastructure (Section 2) that motivate the need for a video control plane (Section 3).
- Using an extrapolation approach to establish the potential room for improvement (Section 4).
- Corroborating these potential gains under a concrete (but simplified) operation model (Section 5 and Section 6).

We discuss outstanding issues in Section 7 and place our work in the context of related work in Section 8 before concluding in Section 9.

2. MOTIVATION

Previous research has confirmed the impact of quality on user experience to show that users are quite sensitive to buffering and high startup latency, and prefer higher bitrate content [3,21]. Given this need for high-quality video delivery, we analyze how today's infrastructure performs.

In this section, we examine the performance of today's delivery infrastructure and highlight potential sources of inefficiencies. We begin by focusing on the end-user streaming video performance. Then, we identify three potential sources of performance problems: variability in client-side, variability within a single ISP or Autonomous System (AS), and variability in CDN performance.

2.1 Dataset

The dataset used in this paper is based on one week of client-side measurements from over 200 million viewing sessions or views (both successful and failed) from over 50 million viewers across 91 popular video content providers around the world. The chosen week includes a single large live event lasting two hours, but otherwise has normal traffic. Table 1 summarizes the data set. The content served by these providers includes both *live* (e.g., sports broadcasts) and *video-on-demand* (e.g., TV episodes and movies). Since we observe similar results from both live and video-on-demand traffic, we show results only on aggregate data from both types of traffic. The data was generated via client-side player instrumentation that collects statistics regarding the current network conditions (e.g., estimated bandwidth to the chosen CDN server) and the observed video quality (e.g., rebuffering ratio, chosen bitrate). Many of the content providers have the option to deliver content to their customers from multiple CDNs; the specifics of how they choose CDNs is proprietary. Due to business and anonymity considerations, we anonymize the providers, CDNs, ISPs, and cities in the following results. Our goal here is to highlight the overall problems in video delivery in general rather than pinpoint inefficiencies of specific ISPs or CDNs.

Time	2011.12.10 - 2011.12.17
Views	281M
Viewers	54M
View hours	30M
Content providers	91
Videos	2M
Countries / Regions	231

Table 1: Dataset Summary

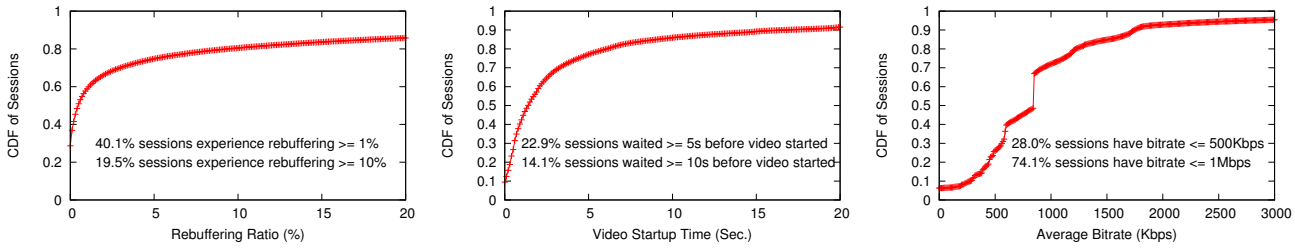
Metrics: We focus on the following industry-standard video quality metrics [6,42]:

- *Average bitrate:* The average bitrate experienced by a view over its lifetime.
- *Rebuffering ratio:* This is computed the buffering time divided by buffering plus playing time, excluding paused or stopped time and buffering time before video start. (We use the term rebuffering ratio and buffering ratio interchangeably.)
- *Startup time:* This is the wait or buffering time before a video starts to play.
- *Failure rate:* This is the percentage of views that failed to start playing and experienced a fatal error during the process. In our experience, these fatal errors usually indicate CDN issues. One example is missing content that the CDN failed to populate to edge servers, and thus users cannot access the video. Another possibility is the CDN server rejecting new connections (e.g., due to overload).
- *Exits before video start:* This is the percentage of views that failed to play the video without experiencing a fatal error. There are generally two causes: (1) users are not interested in the content, and (2) users waited too long for the video to load and lose patience.

We choose these metrics because earlier work showed that they have a significant impact on user engagement [21]. Our goal is not to design an aggregate quality metric that can combine these factors (e.g., [8]). Rather, we want to show the inefficiencies of today's infrastructure and provide directions to improve the video quality. Thus, we consider each metric in isolation in this study.

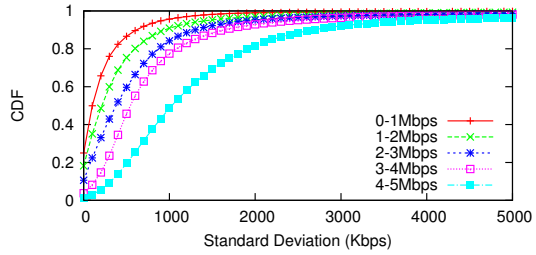
2.2 Video quality today

We begin by analyzing the video quality that today's delivery infrastructure provides before looking at more in-depth analysis to understand reasons for this inefficiency.

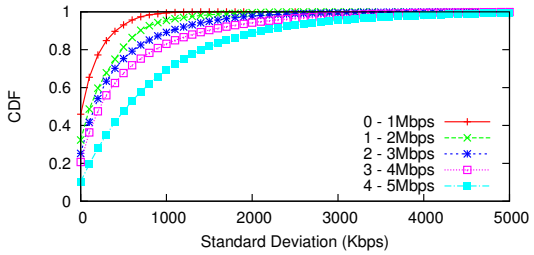


(a) Rebuffering Ratio (b) Video Startup Time (c) Average Bitrate

Figure 1: Distribution of three standard video quality metrics computed over $>$ 200 million user views across 91 providers. The result shows that a non-trivial fraction of views suffer quality issues.



(a) Intra-session



(b) Inter-session

Figure 2: There is significant variability in client-side bandwidth both within and across sessions confirming the need for bitrate adaptation.

Figure 1 shows the distribution of rebuffering ratio, video start up time, and average bitrate from views that have started the video playing. Note that these are the observed performances in the wild with the default video players that the providers use. For rebuffering ratio and average bitrate, we remove sessions less than one minute, because they usually come from users that are not interested in the video.

The result shows that

- 40% of the views experience at least 1% rebuffering ratio, and 20% experience at least 10% rebuffering ratio.
- 23% of the views wait more than 5 seconds before video starts, and 14% wait more than 10 seconds.[1]
- 28% of the views have average bitrate less than 500Kbps, and 74.1% have average bitrate less than 1Mbps.

We also observe that 2.84% views failed to start due to fatal errors, and 14.43% without errors (not shown). Furthermore, we see that more than 9% of the views have actually waited at least 20 seconds before they lose patience in waiting for the video to start.

Implications: To put these results in perspective, previous work shows that a 1% increase in rebuffering ratio can reduce the total play time by more than 3 minutes, viewers who have low join times are more likely to return to the content providers, and viewers who receive higher bitrate videos are likely to watch the video longer [21]. Our analysis indicates that today's end user experience is far from perfect, and highlights the need for performance optimization.

[1]These are the views that have in fact started playing the video.

2.3 Sources of quality issues

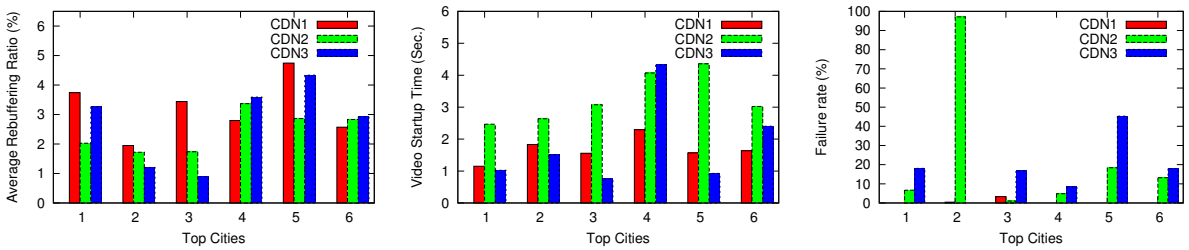
Next we identify and analyze three potential issues that could result in poor video quality.

Client-side variability: Figure 2 shows the distribution of the standard deviation of the client-side intra- and inter-session estimated bandwidth, which shows significant variability in client-side conditions. In this result, we rely on the client player's bandwidth estimation logic which effectively measures the observed bandwidth for the data transferred from the selected CDN server, and the data is collected every 10 seconds. For intra-session bandwidth, we compute the standard deviation of all the bandwidth samples across the entire lifetime of a view. Then we plot the CDF for all views, excluding views that have only one sample. For inter-session bandwidth, for each viewer, we compute the average bandwidth of each session and then compute the standard deviation across the different sessions initiated by that viewer. In both cases, we bin the different views (for intra-session) or viewers (for inter-session) based on their average bandwidth and show the distribution for the five bins from 0-1Mbps to 4-5Mbps. For views with bandwidth less than 1Mbps, more than 20% have a intra-session deviation of 400Kbps. The deviation is 2Mbps for views with bandwidth between 4-5Mbps. Furthermore, there is a fair amount of variability in the inter-session case as well. For example, more than 20% of the viewers with bandwidth less than 1Mbps have a deviation of 250Kbps. We also confirmed that such variability is a general phenomenon that occurs across all ISPs (not shown).

Implications: Given today's bitrate levels (e.g., 400, 800, 1000, 3000 Kbps), this naturally implies the need for intelligent bitrate selection and switching to ensure a smooth viewing experience. Specifically, we see that it is necessary to choose a suitable bitrate at the *start* of each session to account for inter-session variability and also dynamically adapt the bitrate *midstream* to account for intra-session variability.

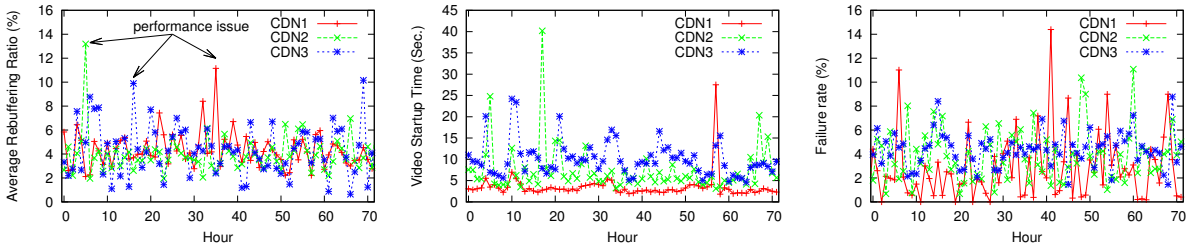
CDN variability across space and time: The performance of CDN infrastructure for delivering video can vary significantly both spatially (e.g., across ISPs or across geographical regions) and temporally. Such variation can be caused by load, misconfiguration (e.g., content not reaching a CDN's edge servers), or other network conditions. Our goal is not to diagnose the root causes of these problems (e.g., [32]), but to show that they occur in the wild.

Figure 3 shows the average rebuffering ratio, video startup time, and video start failure rate experienced by clients with three major CDNs across different geographical regions during the busiest



(a) Rebuffering Ratio (b) Video Startup Time (c) Video Start Failure Rate

Figure 3: CDN performance can vary substantially across different geographical regions



(a) Rebuffering Ratio (b) Video Startup Time (c) Video Start Failure Rate

Figure 4: CDN performance within a given geographical region can vary significantly over time as well

CDN	Rebuffering Ratio	Startup Time	Failure Rate
1	34.25%	79.08%	53.85%
2	25.22%	12.55%	37.50%
3	40.53%	8.37%	8.65%

Table 2: Percentage of scenarios where one of the CDNs performs the best in terms of each of the quality metrics.

hour on a weekday. Here, we choose the geographical regions corresponding to the top six cities by user population. Since there is a potential tradeoff between a session's bitrate and its performance under these quality metrics (higher bitrates will typically result in higher rebuffering ratios), we focus only on sessions having the same bitrate by choosing the most commonly used bitrate within that geographical region. We also remove sessions that cannot sustain the lowest bitrate (300Kbps) to rule out client-side effects in this analysis.

In summary, the results in Figure 3 show that:

- The performance of different CDNs can vary within a given city. For example, in City1, the rebuffering ratio of CDN1 is almost 2× that of users with CDN2.
- For each metric, no single CDN is optimal across all cities. For example, in the case of rebuffering ratio, CDN1 is optimal for City4 and City6, CDN2 for City1 and City5, and CDN3 for City2 and City3.
- CDNs may differ in their performance across metrics. For example, when we consider video startup time, CDN3 performs the best in all cases except City4. In contrast, when it comes to failure rate, CDN3 performs the worst.

Figure 4 shows the same metrics for one of these top cities over three days. (Each point is the average over several thousand sessions.) Here, we see that:

- For all three metrics, no CDN has the best performance all the time. Every CDN experiences some performance issues during the 3-day period. Table 2 shows how often each CDN is the best choice in a city-hour pair over the course of one weekday.[2]
- The rebuffering ratio and failure rate of a CDN may experience high fluctuations over time. For example, for roughly half of the time CDN3 has the lowest rebuffering ratio, and for the other half it has the highest rebuffering ratio.
- Most of the performance degradation is not correlated across CDNs, suggesting that these variations are not merely due to time-of-day effects but other factors.

One possible reason for such variability in the quality observed with CDNs is the load on the CDN. Figure 5(a) shows the rebuffering ratio vs. normalized CDN load for one CDN in one city over a week. Here, we measure the load as the number of unique sessions that we observe over each 5-minute interval. Since our clients represent only a fraction of the total load on the CDN, we normalize the observed load for each CDN by the maximum observed over the entire week for that CDN. Figure 5(a) shows that the rebuffering ratio generally increases with the normalized load.

Implications: This result highlights the need for providers to have multiple CDNs to optimize delivery across different geographical regions and over time. It also suggests that dynamically choosing a CDN can potentially improve the overall video quality.

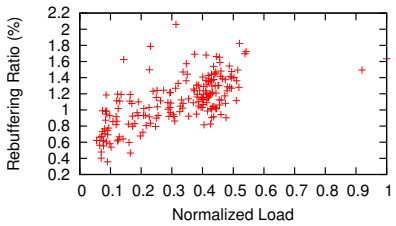
AS under stress: Finally, ISPs and ASes can also experience quality issues under heavy load. Figure 5(b) shows the rebuffering ratio of one AS from all three CDNs during a 4-hour flash crowd period.[3] Each point shows the average buffering ratio across clients at a given time. We report the normalized load on the x-axis by dividing the current number of users by the maximum number of clients observed over time. During this flash crowd, the rebuffering ratio becomes quite high when the number of views increase.

Implications: These results suggest that heavy load can lead to ISP congestion. Ideally, we want the video delivery infrastructure to be

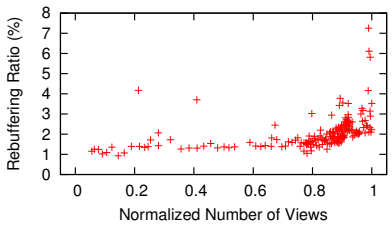
[2]Here we consider only city-hours where all three CDNs serve a reasonable number of views.

[3]This data comes from a known event which is not included in the data set presented before.

aware of these network hotspots to optimize video quality. In this case, the load increases on *all* CDNs and thus switching to a different CDN would not help. As a content provider, one reasonable policy is to reduce the bitrate for all views during these overload scenarios or provide higher quality only to "premium" customers.



(a) One CDN under Load



(b) Whole AS under Load

Figure 5: Rebuffering ratio under stress

2.4 Summary of key observations

The goal of this section was to analyze the current state of video delivery quality and analyze potential sources of performance problems. We see that:

- A significant fraction of sessions suffer quality issues, with more than 20% sessions having 10% rebuffering, and more than 14% sessions with 10 seconds of startup delay.
- There is significant variability in client-side bandwidth both within and across sessions, suggesting the need for intelligent bitrate adaptation.
- CDN quality varies considerably both across time and across space, which indicates the need for providers to dynamically choose different CDNs for different clients.
- When the streaming demand exceeds the capacity of CDNs or/and ISPs, content providers may need to enforce a global policy across clients to ensure a good viewing experience.

3. FRAMEWORK FOR OPTIMIZING VIDEO DELIVERY

The previous section highlighted that many video sessions today observe serious quality issues that arise as a consequence of client-side variability, spatio-temporal variability in CDN performance, and occasionally due to overload. The natural question then is how can we design an optimized video delivery mechanism that is robust to such conditions. In this section, we begin with an overview of the design space of optimizing video delivery and then sketch a high-level vision for a video control plane.

3.1 Design Space

The design space for optimizing video delivery quality has three natural dimensions:

1. *What* parameters can we control?
   There are two main parameters here: choice of bitrate and choice of CDN/server to serve the content. Because the specific video server is controlled by the CDN (e.g., based on load and latency), we only consider server selection at a CDN granularity.
2. *When* can we choose these parameters?
   There are two natural options here. We can select the parameters (i.e., CDN, bitrate) at *startup time* when the video player is launched or dynamically adapt these *midstream* in response to changing network conditions.
3. *Who* decides the values for these parameters?
   There are three high-level options we envision here: purely client-side mechanisms (the de-facto approach today), server-driven mechanisms (e.g., [28]), and an alternative *control plane* that selects these parameters based on global state.[4]

What parameter?	Who chooses?	When to choose?
CDN, Bitrate	Client	Startup
CDN	Client	Startup
Bitrate	Client	Midstream
CDN, Bitrate	Control Plane	Startup
CDN	Control Plane	Startup
Bitrate	Client	Midstream
CDN	Control Plane	Midstream
Bitrate	Client	Midstream
CDN, Bitrate	Control Plane	Midstream

Table 3: Some examples from the overall design space for optimizing video delivery quality. We do not consider the cases where the client chooses the CDN and the control plane chooses the bitrate.

Note that this assumes the viability of two mechanisms—bitrate adaptation and CDN switching—which are already widely used. (The specific algorithms to implement these mechanisms are orthogonal to the focus of this paper.) Bitrate adaptation is already widely adopted in industry (e.g., [1, 4]). Similarly, CDN switching is already adopted in many industry players and with HTTP chunking, chunks can be requested from different CDNs without affecting user experience.

Table 3 looks at some example points in this design space by combining different options of these three variables. At the simplest end of the spectrum (row 1), we can think of a static selection of both CDN and bitrate by the client when the player is launched. This approach is not robust as both changes in CDN performance and client access bandwidth can impact the user experience. The de-facto approach today, shown in the second row in the table, is *client-side* bitrate adaptation but with the CDN/server fixed at start time [1, 4, 7]. There are two advantages of client-side adaptation: (1) clients are in the best position to observe local network effects and (2) the response time to react to network dynamics will be low. As we saw earlier, there is significant temporal and spatial variability in CDN performance that is difficult to detect and alleviate with purely client-side strategies.

To this end, we believe it will be helpful to have a *control plane* deployed either by a content provider or a third party on the behalf of the content providers that is aware of such temporal and spatial variations. (We discuss what such a control plane may look like in the next subsection.) Beyond these performance insights, a control plane also offers content providers more flexibility in instrumenting fine-grained policies; e.g., providing higher quality service to premium customers under load. In the ideal case (last row), we envision this control plane can dynamically adapt both the CDN and bitrate midstream based on global knowledge of network, distribution of active clients, and CDN performance.

[4]We do not consider server-driven mechanisms because these can be equivalently realized by via client- or control-plane mechanisms.

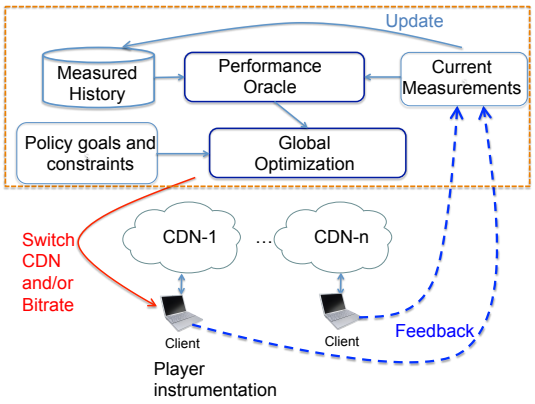


Figure 6: Overview of a video control plane

Of course, it may be unnecessary and impractical for this control plane to be continuously involved in adapting the CDN and bitrate. Thus, we can consider intermediate points in this design space as well. For example, CDN selection can be driven by the control plane because it has a global view of CDN performance, but bitrate adaptation may run purely at the client (rows 4 and 5).

3.2 Vision for a Video Control Plane

The notion of a centralized control plane to optimize content delivery is not new and has been used within CDNs and ISPs for server selection and content placement [12, 13, 36]. There are two key differences in the context of video optimization. First, we introduce a new dimension of *cross CDN* optimization and combining this with bitrate selection/adaptation. Second, we focus on the possibility of *midstream switching* of both parameters, whereas most CDN optimizations focus only on start-time selection. As a simple starting point, our current work assumes that this control plane operates *per content provider*. That is, a video content provider such as YouTube or Hulu runs such a control plane to monitor and improve the video experience for its customers. We discuss issues involving the interaction between multiple such providers and controllers in Section 7.

Figure 6 shows a high-level overview of the three key components in the video control plane: (1) a *measurement* component responsible for actively monitoring the video quality of clients, (2) a *performance oracle* that uses historical and current measurements to predict the potential performance a user will receive for a particular combination of CDN and bitrate at the current time, and (3) the *global optimization* engine that uses the measurement and performance oracle to assign the CDN and bitrate for each user. Next, we briefly highlight the main factors and challenges involved in the design of each component.

Measurement Engine: The measurement engine periodically collects quality statistics for currently active users. Because the client-side player is in the best position to measure the observed video quality, we envision the client player *periodically* (every few seconds) reporting such statistics. In addition to reporting the video quality metrics (e.g., buffering, join time, average bitrate), the measurement engine also collects user and session attributes such as the ISP, location, current CDN being used, and player version that will aid in the performance prediction. The challenge here is to choose a suitable granularity of attributes and quality metrics to measure, and to decide an appropriate frequency at which these reports are sent to the control plane.

Performance Oracle: The performance oracle plays a key role in answering *what-if* style questions at the control plane to predict the performance (e.g., rebuffering ratio, startup delay, failure rate) that a given user may observe at the current time if it chose a different combination of CDN and bitrate. By design, the oracle will have to *extrapolate* the performance based on past and current measurements. For example, it may cluster users based on a set of attributes (e.g., ISP, location) and use the empirical mean within this cluster as its prediction. The challenge here is that the extrapolation must be robust to noise and missing data; e.g., are there enough points within this cluster for this extrapolation to be statistically sound?

Global Optimization: At a high-level, we are solving a *resource allocation* problem, where the resources are the CDNs. Each CDN is characterized by a given network capacity (i.e., how many clients can it serve) and distribution costs. We want to assign each user a suitable CDN and bitrate that maximizes some notion of *global utility* for the content providers and consumers, while operating within the provider's cost constraints and the CDN capacities. There are three main challenges here. First, we want to choose a suitable utility and policy objective. For example, this utility can be a function of the bitrate, quality metrics such as buffering, and the providers' policy goals (e.g., premium customers get higher priority over non-paying users). Designing a good video utility metric that can combine different notions of quality (e.g., bitrate, rebuffering, startup delay) is an open challenge that is outside the scope of this paper [21, 35, 40]. The provider can also specify other policy constraints; e.g., should it admit new clients when all CDNs are overloaded. Our focus is to make a case for such a framework and present initial steps toward a practical realization rather than prescribe specific utility or policy functions. Second, this optimization must fast enough in order to periodically re-optimize the assignments in response to network dynamics. Third, we need to ensure that the optimization is stable and does not itself introduce biases or instability (see Section 5.1).

4. POTENTIAL FOR IMPROVEMENT

Before attempting to design a specific control plane, we want to first establish the improvement in video quality that we can achieve. To this end, in this section we analyze the potential improvement that clients could achieve by choosing a better CDN. As we will see later, the techniques described here can be extended to realize the performance oracle described in the previous section.

4.1 Approach

Our goal is to determine the potential performance improvement assuming each session makes the best possible choice. Ideally, each client will try all possible choices and pick the one with the best performance (e.g., rebuffering rate). Moreover, a client will constantly re-evaluate the performance and switch, if needed, to improve its performance. For example, a client can start with the configuration $(CDN_1, bitrate_1)$, and later switch to $(CDN_2, bitrate_2)$, if the new choice provides better performance. Of course, in practice we cannot have each client continuously probe all possible combinations. To get around this limitation, we *extrapolate* the performance a client could have achieved based on our observed performance of other clients that share similar *attributes*, such as ISP, location, device, and time-of-day. We follow previous work on non-parametric prediction [24, 33] with some simplifying modifications.

We make two simplifying assumptions. First, we do not consider bitrate selection in this section. Second, we assume that session outcomes are independent and that CDN performance does not degrade with load. We relax these assumptions in Section 5.2.

Our approach has two logical stages: *estimation* and *extrapolation* that we describe next.

Estimation: In the estimation step, we compute the empirical performance of each combination of attribute and parameter values.

Let a denote a set of values of a client's attributes, e.g., ISP = *AT&T*, City=*Chicago*, Device=*XBox*. Further, let S_a denote the set of clients sharing same attribute values a, and let $S_{a,p}$ denote the set of clients with attribute values a that have made the same choice or parameter p (i.e., CDN). An example of such set would be, XBox devices of Comcast's subscribers located in Chicago that stream content from Akamai.

For each set $S_{a,p}$, we compute the empirical distribution for the metric of interest, e.g., rebuffering ratio. Let $PerfDist_{a,p}$ denote this empirical distribution. Given two such performance distributions, we say that $PerfDist_{a,p_1}$ is better than $PerfDist_{a,p_2}$, if $\text{MEAN}(PerfDist_{a,p_1}) < \text{MEAN}(PerfDist_{a,p_2})$.[5]

Extrapolation: We use $p_a^* = \text{argmin}_p\{\text{MEAN}(PerfDist_{a,p})\}$ to denote the parameter with the best performance distribution for this specific value of the attribute a. Using this definition, we can extrapolate the best possible performance that can be achieved by a session with attribute values a by selecting parameter p_a^*, and assuming that the performance experienced by the session is randomly drawn from the distribution $PerfDist_{a,p_a^*}$ as shown in Figure 7(a).

Now, for such extrapolations to be statistically meaningful, the number of observations for each a, p setting $|S_{a,p}|$ should be reasonably large. Unfortunately, we run into the classic curse of dimensionality; that is, as the attribute space becomes more fine-grained, the available data becomes sparse [14]. This is particularly problematic because we will be picking the CDN with the highest extrapolated value using this methodology. If we pick the highest among several noisy predictions, we may show improvements even where there are none.

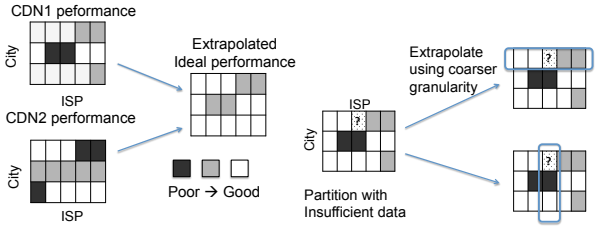
Hierarchical estimation and extrapolation: To address the problem of data sparsity at finer granularity, we use a hierarchical approach [25, 33]. We begin with an exhaustive enumeration of all possible combinations of attributes. That is, if $A = \{a_1 \ldots a_n\}$ is the set of client attributes, we construct the *powerset* 2^A of all attribute combinations. Let $attrset \in 2^A$ denote one such combination of attribute elements such as {isp, location, timestamp}, {isp, location} or {isp}; let a_s denote the values of the attributeset $attrset$ for session s.

Note that a given video session's performance measurement will contribute to multiple attribute partitions corresponding to different granularities. That is, a session with ISP = *AT&T*, City=*Chicago* gets added to the partitions (ISP = *AT&T*, City=*Chicago*), (ISP = *AT&T*), and (City=*Chicago*). Then, for the partitions $S_{a,p}$ that have a sufficient number of data points (we use a threshold of 1000 sessions), we estimate the expected performance using the empirical mean.

The extrapolation step becomes slightly more involved. As discussed earlier, we want to use the most fine-grained attribute information available, but we may not have sufficient statistical confidence in predictions from very fine-grained groups. In such cases, we want to identify a *coarser* attribute combination (see Figure 7(b)) at which we have sufficient data. To get a suitable coarsened granularity, we consider a logical ordering on the powerset of all attribute combinations 2^A such that finer-granularity combinations come earlier. That is, if we had three attributes ISP, city, and timestamp, then {ISP,city,timestamp} < {ISP, city} < {ISP}.[6] Given this ordering, we proceed up the hierarchy from the finer to coarser partitions, until we have sufficient data to make a prediction.

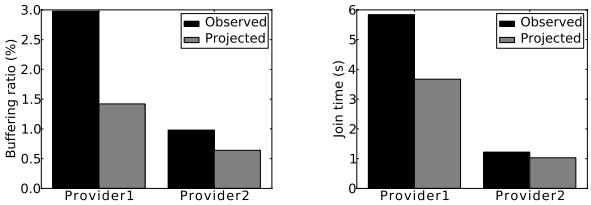
[5]Other possible metrics to compare two distributions can be median, or, more generally, the q-quantile of the distribution.

[6]Strictly speaking this is a partial order. In practice, we break the ties arbitrarily noting that it does not affect the results significantly.

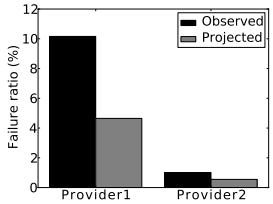


(a) Basic extrapolation (b) Insufficient data

Figure 7: There are two user attributes: ISP and city. For each combination of attribute values, we want to analyze the potential improvement by choosing a better CDN (a). If a combination does not have sufficient data, we identify a suitable coarser level (b). In the simplest case, the extrapolation uses the mean of the distribution within each partition.



(a) Buffering ratio (b) Join time



(c) Failure rate

Figure 8: Potential improvement in three quality metrics (rebuffering ratio, failure rate, and join time) for two providers.

Let a_s^* denote the value of session attributes at this point in the hierarchy. Based on the chosen level in the hierarchy and the parameter setting $(a_s^*, p_{a_s^*}^*)$, we extrapolate the performance by drawing a value from the empirical distribution $PerfDist_{a_s^*,p_{a_s^*}^*}$. In other words, by replacing one performance distribution with another we are simulating the effect of choosing a better parameter setting. As a further refinement, we can also imagine mapping each session into an appropriate percentile bin in the new distribution. That is, if the current session was within the 90-95th percentile of clients in $S_{a,p}$, then we draw from the same bin of 90-95th percentile of the performance of S_{a,p_a^*}. Intuitively, this emulates a policy of not switching unless the quality is likely to improve. For brevity, we do not show the percentile-based extrapolation, noting that this refinement will magnify the potential for improvement.

4.2 Improvement Analysis

For this analysis, we choose two popular providers that use multiple CDNs for video delivery but do not explicitly optimize the CDN based on observed quality or assign clients preferentially to CDNs based on their quality. This ensures that the potential improvement we extrapolate by choosing a better CDN is unbiased.

Validation: We acknowledge that any such extrapolation analysis is challenging and necessarily involves simplifying assump-

tions. That said, we tried to do a careful job in leveraging our large dataset. One specific type of validation we perform is to ensure that we do not spuriously predict improvements when there are none. To this end, we create an artificial dataset by replacing the actual CDN in each session with a different CDN chosen *uniformly at random*. The idea here is that this synthetic dataset should in theory have no scope for improvement (modulo small stochastic effects). We run our algorithm over this synthetic dataset and confirm that our extrapolation predicts negligible (0.05%) improvement.

Average improvement: We begin by computing the average improvement in video quality over a one week period for the two providers using the above extrapolation approach. Figure 8 shows the average improvement for three video quality metrics: buffering ratio, join time, and failure rate. The result shows that for Provider1, we see a significant (more than $2\times$) decrease in the buffering ratio from 3.0 to 1.4. Provider1 also shows significant potential for improvement in the failure rate (10% to 4.5%) and the join time (5.8s to 3.6s). In contrast, the delivery quality for Provider2 is already very good, and thus the scope for improvement there is comparatively lower in the average case. However, as we see next, even Provider2 shows significant improvement under more extreme scenarios.

Improvement under stress: We expect the room for improvement to be significantly higher under more extreme scenarios; e.g., a particular CDN performs poorly in a particular region or has a lot of failures. To this end, we pick specific time segments where we observe large incidents where Provider1 and Provider2 see marked degradation in performance. Then, we analyze the potential for improvement in the video quality under these more extreme scenarios in Table 4 and Table 5 respectively. The results show a dramatic improvement in the potential performance over fairly long durations: $10\times$ for buffering ratio and $32\times$ reduction in failure rate for Provider1 and up to $100\times$ improvement in the failure rate for Provider2.

Main observations: To summarize, our analysis shows that better CDN selection can show marked improvement in video delivery quality. Specifically, we find

- More than $2\times$ improvement in mean buffering ratio and startup time, and $1.6\times$ reduction in failure rate in the average case.
- 10-32$\times$ improvement in the buffering ratio and failure rate over extended time periods under stress.

5. TOWARD A PRACTICAL DESIGN

The previous section establishes that there is a non-trivial potential for improvement. In practice, a control plane has to also take into account the impact of bitrate on performance, effect of CDN load, and also rely on past estimates to predict what the future performance will be. Furthermore, we had also ignored the tractability of global optimization and the specific utility functions or policy objectives.

In this section, we present a preliminary effort at addressing these issues. Our goal is not to realize an "ideal" control plane; rather, we want to establish a *feasible but concrete* control plane design that we can use to analyze if the benefits promised from the previous section can be translated into reality.

5.1 Optimization

To make our discussion concrete, we focus on a specific policy objective and utility function. Our policy goal is to achieve both *fairness* (i.e., do not deny clients if there is sufficient capacity) and *efficiency* (i.e., maximize aggregate utility). There is a rich literature on balancing efficiency-fairness tradeoffs that we can build on here; as a starting point we choose a simple goal of first ensuring fairness and then optimizing for efficiency.

Metric	Duration (hrs)	Current	Projected
Buffering ratio (%)	3	10.41	0.795
Start time (s)	2	6.41	1.997
Failure ratio (%)	1	16.57	0.213

Table 4: Potential improvement in the mean performance under extreme scenario for Provider1

Metric	Duration (hrs)	Current	Projected
Buffering ratio (%)	1	2.24	0.29
Start time (s)	7	1.56	0.39
Failure ratio (%)	3	35.6	0.3

Table 5: Potential improvement in the mean performance under extreme scenario for Provider2

We fix the utility function to be a simple linear combination of the different performance metrics that captures the expected viewing time for a given session. We choose the function

$$Utility = -3.7 \times BuffRatio + \frac{Bitrate}{20}$$

where $BuffRatio$ is in percentage and $Bitrate$ is in Kbps. This utility function is based on prior observations on linear relationship between the expected play time and the different quality metrics reported in a previous measurement study; e.g., a 1% increase in buffering ratio caused a 3.7 minute drop in viewing time [21].

Having fixed the utility function and policy objective, we use a simple two-phase algorithm. First, we assign the clients a *fair share* of the CDN resources by computing the average sustainable bitrate. This allocation ensures that each client has been assigned to some *feasible* combination of bitrate and CDN and there is no unfairness in the allocation in terms of bitrates. Also, each client is assigned a CDN at random so that each CDN gets assigned a share of clients proportional to its capacity. Next, we use this allocation as a starting point and try to incrementally improve the total utility. This proceeds as a greedy algorithm where at each iteration, it picks the combination of client and CDN/bitrate setting that provides the largest incremental contribution to the global utility function.

The intuition behind this two-phase approach is as follows. A pure greedy approach optimizes efficiency (i.e., total utility) but may leave some clients unassigned; e.g., it might initially assign clients with a higher bitrate but may end up saturating the capacity and drop some clients. A pure fair-sharing approach on the other hand guarantees that all clients will be assigned if there is sufficient capacity, but it is agnostic to the performance variability across CDN-client-bitrate combinations. Our approach strikes a balance between these two extremes by ensuring all clients are assigned, and improves the efficiency from this starting point. We do not claim that this optimal in a theoretical sense but we do observe that it works well across a range of realistic workloads.

One subtle issue is that the optimization may itself introduce undesirable temporal biases. For example, if we discover that CDN1 is performing poorly and shift all clients to CDN2, then it impacts our ability to predict the performance of both CDN1 (we do not have any samples) and CDN2 (we have increased its load) in the future. This is a classical "exploration-exploitation" tradeoff (e.g., [20]). We face a particularly difficult form of this problem; our rewards are not independent (due to CDN load), and the characteristics of CDNs change over time even if we do not use them. However, we do have access to a large amount of data. A simple solution in this case is to use some form of randomization. Here, we choose a random subset of sessions that will not receive any explicit optimization so that we can observe their performance in

the wild. With a large enough population even a small fraction (say 2-5%) of unoptimized clients would suffice to build a robust prediction model. There are other more sophisticated approaches to solve this problem, such as knowledge gradient algorithms [38], that will further reduce such biases. We currently use the simple randomization approach due to its ease of implementation.

5.2 Performance estimation

A key issue with performance extrapolation we already observed in Section 4 is the need for a prediction mechanism that is robust to data sparsity and noise at such finer granularities. We address this by building on the hierarchical extrapolation techniques described in Section 4. There are three additional practical challenges that need to be addressed here.

First, in Section 4, our goal was to estimate the potential for improvement. Hence, we assumed access to a *performance oracle* that has a current view of the performance. In practice, a real control plane will not have such an oracle and will have to rely on historical measurements. We extend the hierarchical approach from the previous section to make predictions based on recent historical measurements of the performance for specific CDN-client-bitrate combinations. Since CDN performance also shows significant temporal variability, we simply use the most recent information (e.g., last hour). In Section 6.3, we show that even this simple use of historical measurements works very well in practice.

Second, the extrapolation in the previous section ignores the effect of CDN load and how the performance degrades as a function of load. To this end, we augment the performance estimation step to also model the CDN load. Specifically, we observe that the CDN performance shows a roughly thresholded behavior where the performance is largely independent of the load up to some threshold T_1, after which the performance degrades roughly linearly, and at a higher load threshold T_2, the performance would drop significantly. We select these thresholds based on observed measurements (not shown for brevity).

Third, we did not consider bitrates in the previous section. Here, we simply treat bitrate as an additional attribute to our decision process. That is, in addition to characteristics such as ISP and city, we also partition clients based on their current bitrate when building the performance estimators and prediction models. We extend the measurements from the previous section to reflect bitrates in our prediction model.

6. TRACE-DRIVEN SIMULATIONS

In this section, we use trace-driven simulations to evaluate the qualitative benefits of the global control plane approach we instantiated in Section 5 over other choices in the design space from Table 3 under different scenarios.

6.1 Setup

We built a custom simulation framework that takes the following configurations as input: (1) client arrival and viewing time patterns obtained from measurements from the same dataset described in Section 2, and (2) empirically observed CDN performance distribution in different geographical regions at different (normalized) loads. In each epoch, a number of clients join the system and choose a viewing time drawn from empirical distribution. The client either stays until the end of viewing time, or leaves when its utility becomes 0, i.e., the performance becomes unbearable. In this simulation, we implement three strategies:

1. *Baseline:* Each client chooses a CDN and bitrate at random. This can be viewed as a strawman point of comparison.
2. *Global coordination:* The control plane algorithm proposed in Section 5.
3. *Hybrid:* Each client is assigned to a CDN with lowest load and a bitrate by the global optimization when it first arrives, but the subsequent adaptation is only limited to client-driven bitrate adaptation, i.e., no midstream CDN switching. This can be viewed as a simplified version of what many providers deploy today: start time CDN selection and client-side mid-stream bitrate adaptation.

The primary goal of these simulations is to analyze the qualitative benefits of a global control plane. These alternative points do not necessarily reflect exact strategies in operation today; we pick these as sample points from our design space in Table 3.

In each epoch, after the decisions are made, the simulator computes the resulting load on each CDN (by summing up bitrates from all clients), and then computes the expected performance of each client based on empirical measurements and the resulting load, as described in the previous section. In this simulation, we use three CDNs. The performance metrics of each CDN under normal load are obtained by taking its mean performance over a week, so that any overload effect is averaged out. Then the load thresholds for modeling the impact of CDN load on performance (see Section 5.2) are extrapolated from visually correlating the relationship between the load and quality metrics for each CDN. In order to scale our simulation framework, we normalize the CDN capacity and the number of clients proportionally compared to the actual capacity and number of clients. The normalization ensures that 1) the required capacity to serve the highest bitrate for all clients will exceed the combined capacity of all three CDNs, and 2) the required capacity to serve the lowest bitrate for all clients does not exceed that of any two CDN combined.

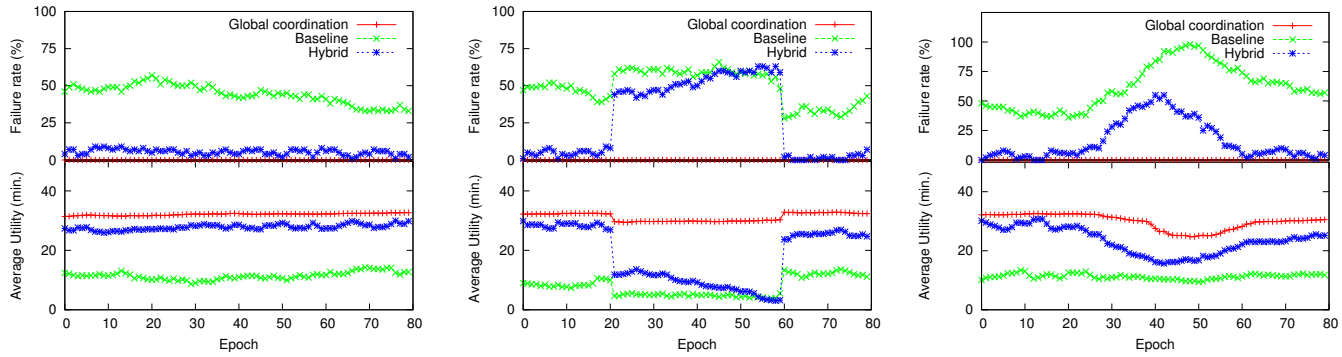
We simulate three scenarios: average case, CDN performance degradation, and flash crowd. In the average case scenario, arrival patterns and playback duration mimic the typical client arrival patterns observed in the dataset. In the CDN performance degradation scenario, we retain the same client arrival pattern, but emulate a sudden degradation in one of the CDNs. Finally, in the flash crowd scenario, a large number of clients arrive simultaneously.

6.2 Results

We focus on two metrics in our evaluation: *average utility* and *failure ratio*. Average utility is computed by the total utility divided by all clients in the system (including clients that failed to play video and thus has zero utility). Since the arrival pattern is the same for all three strategies, they all have the same denominator in all epochs. Failure ratio is the ratio of clients that could not receive any video due to either CDN exceeding capacity or unbearably high rebuffering ratio. Figure 9 shows the simulated time series of these two metrics for the three scenarios across different adaptation strategies.

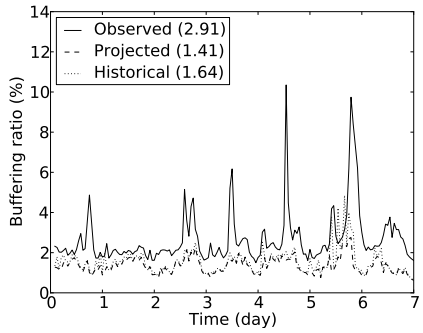
Average case: In Figure 9(a), we observe that global coordination significantly outperforms the baseline strategy in terms of the average utility. One interesting observation is that in the common case, a hybrid strategy provides similar performance to global coordination. This is because in this strategy, a client chooses an ideal CDN-bitrate combination when it arrives, and under regular conditions, this strategy can maintain good performance. Unsurprisingly, the baseline approach performs poorly even in the common case and has both a high failure rate and low overall utility. The primary reason is that this approach is agnostic to CDN performance.

With regard to failure rates, we see that the baseline approach is consistently high. The capacity-aware global coordinator is able to assign clients to suitable CDNs and bitrates and keep a zero failure

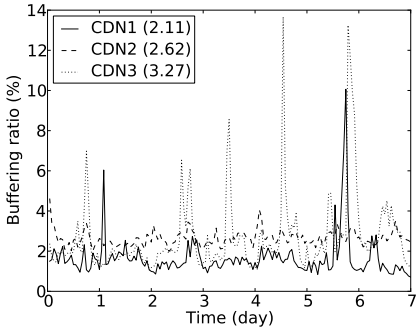


(a) Common case (b) CDN degradation (c) Flash crowd

Figure 9: Simulation results of three scenarios. For each scenario, we show the performance of baseline, hybrid and global coordination in terms of failure ratio and average utility value. (The utility metric is in units of expected minutes of playing time.)



(a) Prediction vs. Optimal



(b) Performance of different CDNs

Figure 10: Performance gap between a history-based and an oracle performance estimator. We also show the performance of the individual CDNs to visually confirm why history-based estimates work in most cases and highlight the specific epochs where it does not.

rate. The hybrid strategy has a small but non-zero failure rate; the reason is that some clients may use high bitrates so that a small number of clients are rejected.

CDN variability: In this scenario, in epochs 20–60, the previously best CDN experiences a huge degradation with the average rebuffering ratio going to 13%, before eventually recovering at epoch 60. Figure 9(b) shows how different strategies respond to CDN variability. In this case, global coordination still maintains zero failure rate (which is possible because of our normalization) and has a significantly higher average utility than both other approaches. However, it too suffers a mild drop in utility compared to the normal scenario during the degradation. We also see that the hybrid approach does almost as poorly as the baseline in both metrics. This is because in this strategy, the server is not aware of CDN performance degradation, and even when the clients can identify this degradation, it is unable to switch CDN midstream.

Flash crowd: Last, we emulate a sudden flash crowd in Figure 9(c), where a large number of clients try to join the system between epochs 20–60. In this case, the global control algorithm lowers the bitrate for many users in order to accommodate more new users. Recall that this is one of the policy decisions we imposed in our design to balance fairness in allocating users vs. efficiency. During the flash crowd, we see that the failure rate for global coordination remains at zero, whereas the baseline and hybrid have exceedingly high failure rates. The hybrid approach does not degrade as much as in previous case because the it is aware of the load on each CDN, but without midstream CDN switching, the performance is worse than global coordination.

6.3 History vs. Oracle prediction

As discussed earlier, a practical control plane will need to rely on historical estimates of the performance to choose the best CDN. A natural question is how far away from the optimal performance is such history-based prediction. To analyze this gap, we use the same dataset for Provider1 from Section 4 and compare a history-based vs. oracle prediction in Figure 10(a) over one week. We consider two options for using the historical estimates: using either the *previous hour* or using the *previous day*. That is, for each partition, we identify the best performing CDN using the previous hour or day measurements and use that as the prediction for the current epoch. The oracle uses predictions based on the *current hour*. The result shows that using the previous hour's predictions, the gap between the history and oracle predictor is small for many instances. However, we also see some cases where there is a non-trivial difference between the historical and optimal. Figure 10(b) visualizes how the individual CDNs' performance varies during this period to provide some intuition on why such large gaps occur. We do see that the CDNs' performance is largely predictable using the recent history but does occasionally spike. These performance spikes cause causes our estimate using the previous hour to become inaccurate. Our preliminary results (not shown) suggest that some of these gaps can be alleviated this using more fine-grained historical information (e.g., previous five minute epochs).

6.4 Summary of results

- Global control plane works well in all scenarios, including CDN performance variation and flash crowd.
- A hybrid approach of using the coordinator only at startup time and relying on pure client-side adaptation may work quite well in common scenarios.
- However, such a hybrid approach could suffer performance degradation under CDN variability and flash crowds. In such cases, a control plane can implement more flexible policies. For example, under flash crowd it maintains a zero failure rate by reducing all client bitrates.
- The benefits of a control plane can be realized with simple extrapolation by using predictions from previous epochs.

7. DISCUSSION

Next, we present preliminary insights on how we can address issues such as scalability, the interactions between such a control plane and CDNs, and interactions across multiple such controllers.

Scalability: A concern with global optimization is scalability vs. number of clients and the time to respond to network events. Our unoptimized implementation in Java takes $\approx$ 30s to run the global optimization for $10,000$ clients, 4 CDNs, and 5 bitrates. We speculate that typical video utility functions will possess a natural diminishing property [23]. Intuitively, this means that the incremental utility in going from a 10% buffering ratio to a 5% buffering ratio will be higher than the increment going from 6% to 1%. In this case, there are known techniques to speed up the greedy step via "lazy evaluation" [30]. Beyond such algorithm optimizations, we also envision scaling the control plane by logically partitioning different geographical regions and running one instance per region.

Switching tolerance: A natural question is how much bitrate switching can users tolerate? Controlled studies suggest users are sensitive both to frequent switches (e.g., [18]) and also to sudden changes in bitrate (e.g., [35]). We do not, however, have a good quantitative understanding on the tradeoff between switching vs. the desire to maintain high bitrate and low buffering. As this tradeoff becomes clearer with future measurement studies, this can be incorporated into the control plane optimization function as well.

Interaction with CDNs: One question is whether CDNs can do (are doing) such optimizations themselves today. While we cannot speculate about their internal policies, measurement studies suggest that CDNs are largely optimizing for latency [27]. Furthermore, content providers increasingly use multiple CDNs and thus no single CDN can provide the required cross-CDN optimization. We do note, however, that the techniques we develop apply equally well in the context of individual CDNs.

Another concern is whether there can be undesirable interactions between such higher-level optimization and CDNs' optimizations. Of particular concern are possible oscillations caused by such interactions. Unfortunately, it is hard to answer this question due to the limited visibility we have into CDN policies. Nonetheless, we hope that this potential problem will be alleviated in the future, as we envision new generation architectures where CDNs expose APIs to content providers and such controllers. For example, more fine-grained information on the available capacity of the CDNs or current load for different geographical regions can inform better control plane optimization strategies.

Finally, an emerging direction is the concept of federated CDNs [19, 37]. A federated CDN incorporates a technology integration and business partnership between carriers and CDNs to provide a unified CDN offering that has a global presence and benefits to both CDNs and content providers. For example, a federated CDN eliminates the need for the content provider to publish to each CDN. The global coordinator proposed in this paper is complementary to a federated CDN and can be used to enable high quality video distribution across a federated CDN. In fact, we believe a coordinator is essential to delivering high quality in a federated CDN.

Multiple controllers: So far, we implicitly assumed a simple model, in which the different controllers are independent, and that one controller's decisions will have limited impact on others. In the future, we expect such controllers to expose APIs to exchange performance data and policy constraints/preferences, similar to the way ISPs use BGP to coordinate.

8. RELATED WORK

Client side measurements: The variability in client-side throughput–across ISPs, within a given viewing session and across multiple sessions–have been well documented in past measurement studies (e.g., [26, 42]). The natural solution in a video streaming context is to adapt the video bitrate in response to changing bandwidth conditions to ensure an uninterrupted viewing experience.

Client-side adaptation: Several commercial products today perform some form of client-side adaptation to adapt to changing bandwidth conditions (e.g., [1,4]) and there are ongoing standardization efforts in this respect [7]. The key difference here is that these focus purely on bitrate adaptation. Recent analysis of commercial players suggest that there is room for improvement in client-adaptation strategies [17, 39]. As we saw in earlier sections, there is significant variability in network and CDN performance [31]. Furthermore, there is an inherent need for coordination under overload which means that even near-ideal client-side mechanisms will not be sufficient. A global control plane can alleviate these concerns by coordinating actions across multiple viewers.

Video Coding: Layered coding and multiple description coding offer alternatives for graceful degradation of video quality (e.g., [15, 34]). While these are attractive in theory, they impose significantly higher complexity on the provider, delivery, and player infrastructure. We do note that if these solutions do get deployed, a video control plane is well-positioned to leverage the additional flexibility that these offer as it can more smoothly degrade performance instead of having to choose from a discrete set of bitrates.

CDN and server selection: Server selection strategies within a CDN are based on proprietary algorithms, but measurement studies suggest that these are largely based on proximity and latency (e.g., [27]). Similarly, in the context of video delivery, the details of how particular providers choose CDNs or direct clients to different CDNs are proprietary. Preliminary measurements, however, suggest that the strategies are largely statically configured (e.g., [41]) and that there appears to be no concerted effort to choose CDNs either at startup (e.g., [10]) or midstream (e.g., [11]). In making a case for a global video control plane, our goal is not to pinpoint the inefficiency of particular providers' selection strategies. Rather, we want to design a general framework for high-quality video delivery.

Other video measurements: Given the growing dominance of video traffic, there have been many measurement studies of deployed systems that focus on: content popularity and access patterns (e.g., [16]), the user's desire for high quality and how it impacts play time (e.g., [21]), user viewing patterns (e.g., [9,22]), and extreme scenarios such as flash crowds (e.g., [43]). These works have been instrumental in exposing performance bottlenecks and implications of user behavior on system design. However, these do not directly focus on optimizing video quality by intelligent choice of CDNs and bitrates, which is the focus of our work.

9. CONCLUSIONS

User expectations of high quality video delivery—low buffering, low startup delays, and high bitrates— are continuously rising. While HTTP-based adaptive streaming technologies have dramatically decreased the barrier for content providers to reach a wide audience, the network and delivery infrastructure these rely on is fundamentally unreliable. Our measurements from over 200 million sessions confirm that this is indeed the case: more than 20% of sessions suffer quality issues such as more than 10% buffering or more than 5 seconds startup delay.

Our motivating question was whether it is possible to deliver high-quality video in such a dynamic environment. Given the significant variability in ISP and CDN performance, we argued the case for a video control plane that uses measurement-driven performance feedback to dynamically adapt video parameters such as the CDN and bitrate to improve the video quality. We established the potential for improvement using measurement-driven extrapolation and find that optimal CDN selection can improve the buffering ratio by up to $2\times$ in normal scenarios and more than $10\times$ under more extreme scenarios. We further augmented these results using trace-driven simulations and confirm the potential benefits of such a control plane.

In making a *case for* a video control plane, our work follows in the spirit of approaches for CDN and ISP management that show the benefit of network-wide views. There are several challenges that need to be addressed before these benefits can be realized in practice: scalability, interaction with CDNs, issues surrounding multiple providers and controllers among others.

Acknowledgments

We thank our shepherd Sujata Banerjee and the anonymous reviewers for their feedback that helped improve the final version of the paper. We also thank Ganesh Ananthanarayanan and Justin Ma for providing comments on early drafts.

10. REFERENCES

[1] Akamai HD Adaptive Streaming. http://wwwns.akamai.com/hdnetwork/demo/index.html.

[2] Cisco forecast. http://blogs.cisco.com/sp/comments/cisco_visual_networking_index_forecast_annual_update/.

[3] Driving Engagement for Online Video. http://events.digitallyspeaking.com/akamai/mddec10/post.html?hash=ZDlBSGhsMXBidnJ3RXNWSW5mSE1HZz09.

[4] Microsoft Smooth Streaming. http://www.microsoft.com/silverlight/smoothstreaming.

[5] Move networks. http://www.movenetworks.com/.

[6] Video quality metrics. http://www.akamai.com/html/solutions/stream_analyzer.html.

[7] I. Sodagar. The MPEG-DASH Standard for Multimedia Streaming Over the Internet. *IEEE Multimedia*, 2011.

[8] K. Chen, C. Huang, P. Huang, C. Lei. Quantifying Skype User Satisfaction. In *Proc. SIGCOMM*, 2006.

[9] L. Plissonneau and E. Biersack. A Longitudinal View of HTTP Video Streaming Performance. In *Proc. MMSys*, 2012.

[10] V. K. Adhikari, Y. Chen, S. Jain, and Z.-L. Zhang. Where Do You Tube? Uncovering YouTube Server Selection Strategy. In *Proc. ICCCN*, 2011.

[11] V. K. Adhikari, Y. Guo, F. Hao, V. Hilt, , and Z.-L. Zhang. A Tale of Three CDNs: An Active Measurement Study of Hulu and Its CDNs. In *Proc. IEEE Global Internet Symposium*, 2012.

[12] K. Andreev, B. M. Maggs, A. Meyerson, and R. Sitaraman. Designing Overlay Multicast Networks for Streaming. In *Proc. SPAA*, 2003.

[13] D. Applegate, A. Archer, V. Gopalakrishnan, S. Lee, and K. K. Ramakrishnan. Optimal Content Placement for a Large-Scale VoD System. In *Proc. CoNext*, 2010.

[14] R. E. Bellman. Adaptive control processes: A guided tour. Princeton University Press.

[15] J. Byers, M. Luby, and M. Mitzenmacher. A digital fountain approach to asynchronous reliable multicast. *IEEE JSAC*, Oct. 2002.

[16] M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, and S. Moon. I Tube, You Tube, Everybody Tubes: Analyzing the World's Largest User Generated Content Video System. In *Proc. IMC*, 2007.

[17] L. D. Cicco and S. Mascolo. An Experimental Investigation of the Akamai Adaptive Video Streaming. In *Proc. USAB*, 2010.

[18] N. Cranley, P. Perry, and L. Murphy. User perception of adapting video quality. *International Journal of Human-Computer Studies*, 2006.

[19] D. Rayburn. Telcos and Carriers Forming new Federated CDN Group called OCX (Operator Carrier Exchange). June 2011. StreamingMediaBlog.com.

[20] D. P. de Farias and N. Megiddo. Exploration-Exploitation Tradeoffs for Experts Algorithms in Reactive Environments. In *Proc. NIPS*, 2004.

[21] F. Dobrian, V. Sekar, A. Awan, I. Stoica, D. A. Joseph, A. Ganjam, J. Zhan, and H. Zhang. Understanding the impact of video quality on user engagement. In *Proc. SIGCOMM*, 2011.

[22] A. Finamore, M. Mellia, M. Munafo, R. Torres, and S. G. Rao. Youtube everywhere: Impact of device and infrastructure synergies on user experience. In *Proc. IMC*, 2011.

[23] G. Nemhauser, L. Wosley, and M. Fisher. An analysis of the approximations for maximizing submodular set functions. *Mathematical Programming*, 14:265–294, 1978.

[24] A. George, W. B. Powell, S. R. Kulkarni, and S. Mahadevan. Value function approximation using multiple aggregation for multiattribute resource management. http://www.scientificcommons.org/53756787, 2009.

[25] I. Ryzhov and W. B. Powell. Bayesian Active Learning with Basis Functions. In *Proc. IEEE Workshop on Adaptive Dynamic Programming and Reinforcement Learning*, 2011.

[26] C. Kreibich, B. N. V. Paxson, and N. Weaver. Netalyzr: Illuminating The Edge Network. In *Proc. IMC*, 2010.

[27] R. Krishnan, H. V. Madhyastha, S. Jain, S. Srinivasan, A. Krishnamurthy, T. Anderson, and J. Gao. Moving Beyond End-to-End Path Information to Optimize CDN Performance. In *Proc. IMC*, 2009.

[28] L. De Cicco, S. Mascolo, and V. Palmisano. Feedback Control for Adaptive Live Video Streaming. In *Proc. of MMSys*, 2011.

[29] H. Liu, Y. Wang, Y. R. Yang, A. Tian, and H. Wang. Optimizing Cost and Performance for Content Multihoming. In *Proc. SIGCOMM*, 2012.

[30] M. Minoux. Accelerated Greedy Algorithms for Maximizing Submodular Set Functions. In *Proc. of 8th IFIP Conference, Springer-Verlag*, 1977.

[31] M. Venkataraman and M. Chatterjee. Effects of Internet Path selection on Video QoE. In *Proc. MMSys*, 2011.

[32] A. Mahimkar, Z. Ge, A. Shaikh, J. Wang, J. Yates, Y. Zhang, and Q. Zhao. Towards Automated Performance Diagnosis in a Large IPTV Network. In *Proc. SIGCOMM*, 2009.

[33] A. K. Mccallum. Learning to use selective attention and short-term memory in sequential tasks. In *Proc. Conference on Simulation of Adaptive Behavior*, 1996.

[34] S. McCanne, M. Vetterli, and V. Jacobson. Low-complexity video coding for receiver-driven layered multicast. *IEEE JSAC*, Aug. 1997.

[35] R. K. P. Mok, E. W. W. Chan, X. Luo, and R. K. C. Chang. Inferring the QoE of HTTP Video Streaming from User-Viewing Activities . In *Proc. SIGCOMM W-MUST*, 2011.

[36] R. S. Peterson and E. G. Sirer. Antfarm: Efficient Content Distribution with Managed Swarms. In *Proc. NSDI*, 2009.

[37] R. Powell. The Federated CDN Cometh. May 2011. TelecomRamblings.com.

[38] I. Ryzhov, P. Frazier, and W. Powell. The knowledge gradient algorithm for a general class of online learning problems. http://www.princeton.edu/~iryzhov/journal/online7.pdf, 2011.

[39] S. Akhshabi, A. Begen, C. Dovrolis. An Experimental Evaluation of Rate Adaptation Algorithms in Adaptive Streaming over HTTP. In *Proc. MMSys*, 2011.

[40] H. H. Song, Z. Ge, A. Mahimkar, J. Wang, J. Yates, Y. Zhang, A. Basso, and M. Chen. Q-score: Proactive Service Quality Assessment in a Large IPTV System. In *Proc. IMC*, 2011.

[41] R. Torres, A. Finamore, J. R. Kim, M. Mellia, M. M. Munafo, and S. Rao. Dissecting Video Server Selection Strategies in the YouTube CDN. In *Proc. ICDCS*, 2011.

[42] M. Watson. HTTP Adaptive Streaming in Practice. http://web.cs.wpi.edu/~claypool/mmsys-2011/Keynote02.pdf.

[43] H. Yin, X. Liu, F. Qiu, N. Xia, C. Lin, H. Zhang, V. Sekar, and G. Min. Inside the Bird's Nest: Measurements of Large-Scale Live VoD from the 2008 Olympics. In *Proc. IMC*, 2009.