

Towards Improved DASH Adaptation in NDN: An Emulative Analysis

Denny Stohr[‡], Timo Kalle^{*}, Andreas Mauthe[§], Amr Rizk[‡], Ralf Steinmetz[‡], Wolfgang Effelsberg^{*}
Technische Universität Darmstadt

^{*}Distributed Multimedia Systems, {wolfgang.effelsberg}@cs.tu-darmstadt.de

[‡]Multimedia Communications Lab, {denny.stohr, amr.rizk, ralf.steinmetz}@kom.tu-darmstadt.de

[§]Information Centric Communication Systems, a.mauthe@lancaster.ac.uk

Abstract—The Information-Centric Networking (ICN) paradigm is deemed to enable simpler and more efficient networking interaction by moving from a strict connection-based relationship between client and server to an interest-based relationship between user and content. Hence, addressing shifts to content objects rather than any specific copy or location of the content. In order to achieve this new way of addressing the client uses the content name, which is propagated into the network. As a result, a higher efficiency is expected since ICN network nodes (e.g., routers) may reply to such requests using copies from their own caches, or different sources hosting a content object may be used in parallel. Thus, this concept promises better support for device mobility and implicit caching, and it provides inherent multicast support.

However, the simplicity of this concept comes at a price. As established applications, such as adaptive bitrate video streaming, have been designed and optimized having a client-server networking environment in mind. In particular, the quality adaptation algorithms used by today's de-facto streaming standard Dynamic Adaptive Streaming over HTTP (DASH) uses bandwidth estimation techniques that are specifically designed for a client-server networking environment where all video segments are retrieved from the same host.

This paper addresses issues related to adaptive video streaming in an ICN environment. It analyzes the video streaming behavior of state-of-the-art video quality adaptation algorithms such as PANDA and BOLA in emulated ICN environments. The analysis focuses on the impact ICN chunk-based throughput measurements and ICN caches have on quantitative measurements of Quality-of-Experience (QoE). The paper provides a detailed investigation of the chunk-based throughput estimation showing that it behaves fundamentally different due to ICN caching. Based on these results we provide extensions to existing adaptation algorithms (e.g., PANDA) that can significantly improve the QoE in ICN environments.

Index Terms—Named Data Networking, DASH

I. INTRODUCTION

Constantly increasing traffic demands as well as device mobility pose significant challenges for Over-the-Top streaming providers such as Netflix and YouTube. Traditionally these technical demands are being solved by employing a Content Delivery Network (CDN). Nonetheless, rising costs, capped bandwidths, and higher video quality (i.e., 1080p, UHD, HDR) together with the users' higher service quality expectations make effective bandwidth use and mobility support crucial considerations in the design of video streaming systems.

Due to the limitations of the host-centric design of the current Internet we reconsider the underlying networking paradigm. In

particular content- and network-centric paradigms are being explored [1]. The underlying idea is that users are interested in the content (i.e., the video segments) rather than any particular copy or location. Hence, the content itself is made a primary object of network transfer, addressing and retrieval. Research has been focusing on the concept of Named Data Networking (NDN) [26], which is an instance of the more general Information-Centric Networking (ICN) paradigm. It supports device mobility and caching as central aspects of the overall networking architecture.

Separating content requests from the explicit knowledge of host addresses and shifting the responsibility of content discovery to the network brings many advantages for dynamic adaptive streaming over HTTP (DASH), such as: i) more robust transitions between networks during streaming sessions [23], ii) support for caching at each forwarding entity in the network to fulfill the demand more efficiently when compared to dedicated CDN nodes for each service [7], iii) network coding to efficiently combine replies by multiple content stores [16], iv) and the ability to implement content-aware forwarding and caching mechanisms to improve the overall efficiency of the network. Yet, with such fundamental changes in the underlying networking concepts challenges arise for existing services such as DASH since they rely directly on a host-bound network architecture. In particular, the DASH adaptation depends on the measurement history and assumes host-bound connections. Since this is not provided in NDNs it can lead to problematic side effects when using DASH in an NDN context. Most crucially, stalling may occur when segments are not available in the caches [14]. However, to our knowledge, there is currently no research sufficiently investigating the effects on the performance of existing DASH adaptation algorithms when running on top of an NDN.

In this paper, we provide a systematic study of the impact of an NDN system on standard DASH adaptation algorithms. To this end i) we propose an NDN evaluation architecture making use of established emulation concepts within mininet, ii) we show that implicit caching in NDN leads to undesirable adaptation effects using traditional DASH adaptation, iii) and we propose a set of extensions for using *any* throughput-based adaptation algorithm in NDN providing an overall improved QoE based on extended segment information in Media Presentation Descriptions (MPDs) and NDN-based throughput

measurements on NDN chunk granularity.

This paper is structured as follows: first, an overview of related work in DASH adaptation algorithms and their evaluation in NDNs is given. Then, the system setup for this study is introduced, followed by a preliminary analysis of DASH in NDN. Subsequently, we present the approach for extending DASH adaptation algorithms for NDN, followed by another evaluation and conclusions.

II. RELATED WORK

In the following, a research overview in the domain of DASH and NDN-specific studies for video streaming and emulation environments for NDN is given, corresponding to the overview provided in Table I.

A. DASH Mechanisms

Much of the focus in DASH research lies on the improvement of the dynamic adaptation behavior, which has resulted in a variety of different approaches [24]. These can be broadly divided into three different categories, namely rate-based, buffer-based and hybrid adaptation.

Rate-based (or throughput-based) adaptation algorithms rely on the estimation of the currently available bandwidth. They compute the throughput of the present connection using a function $f(S_t, \dots, S_{t-n})$ executed after the download of a segment S considering a sliding window of the n previous measurement periods (e.g., by using the Segment Fetch Time or a "Probe and Adapt"-scheme). According to the estimated bandwidth, the quality level is adjusted. Overall, the best choice for the quality level is the highest video bitrate that fits completely within the estimated bandwidth. Ideally, the bandwidth estimation function detects changes in bandwidth fast enough for the algorithm to react and adjust the quality level (e.g., before stalling events occur when the bandwidth drops heavily) while not introducing "adaptation hysteresis" caused by short-term bandwidth fluctuations. PANDA [13] uses a "Probe and Adapt" scheme to predict the bandwidth and to avoid an unfair share of bandwidth at a network bottleneck.

Buffer-based adaptation algorithms use the buffer fill level of the video player (the client) to get feedback about the bandwidth situation. A consistently growing buffer level is an indication that the current playback bitrate is lower than what is provided in the network. If the buffer level falls, the playback bitrate is too high considering the network conditions and should be reduced. The video bitrate is appropriate when the buffer value is balanced and remains more or less constant over time. BOLA [22] uses Lyapunov optimization to calculate the best joint utility to achieve a minimum of rebuffering along with a maximized video quality.

Hybrid adaptation algorithms use a bandwidth estimation function as well as the buffer fill level to capitalize on the advantages from both types of adaptation. Yet, the combination of the direct feedback from bandwidth estimation and the indirect feedback from the buffer observation leads to a highly complex adaptation algorithm. A well-known hybrid algorithm is SQUAD [25].

Irrespective of their category, every adaptation algorithm is based on the trade-off between providing the highest possible quality level, keeping the number of quality switches low and—most important—entirely avoiding stalling events. While aggressive algorithms follow the bandwidth trend closely and have a better chance to provide a higher average quality level, they have a higher switching frequency (due to their reaction to short positive and negative bandwidth spikes), and they are more prone to stalling events. On the other hand, conservative algorithms have a smoother behavior and ensure a stable and solid video quality with the downside that the actually provided quality level is not the highest possible one.

B. NDN Mechanisms

With the emergence of ICN the issue of video streaming, and particularly DASH needed to be addressed in a way that would allow dealing with the specific conditions introduced by the content- and information-centric paradigm. The first publications related to this topic were experimental studies of video streaming over CCN. Here, [2, 14] explore how specific varying configurations and parameters, e.g., the overlay path and the chunk size, affect the bandwidth usage while streaming.

In addition, the caching problems are further investigated. In-network caching often leads to video quality oscillations because with standard DASH a cache hit will be interpreted as an improvement in network conditions, which is consequentially followed by an adaptation to a higher video quality. This change can then result in cache misses if the next segment is not available at the cache, followed by an adaptation back to the lower video quality. In order to deal with this [8] implements an in-network mechanism, i.e., in every cache only the segment with the highest bitrate is stored in the cache, meaning that lower bitrate segments are replaced. Also, a transcoding mechanism is attached to the cache so that every interest requesting the same or a lower bitrate can be satisfied by, if necessary, transcoding the higher bitrate to the requested bitrate. A recent publication systematically compares DASH in NDN and TCP [21]. The authors implemented PANDA and BOLA to cover rate-based adaptation as well as buffer-based adaptation. The results are that the basic version of an NDN cannot keep up with TCP when DASH is used. An extension for NDNs for in-network loss recovery (e.g., Wireless Loss Detection and Recovery) is proposed that performs at the same level as TCP. With a load balancing mechanism for NDN (plus this in-network loss recovery), their results show that NDN can outperform TCP. Furthermore, they use a finer-grained bandwidth estimation on the NDN chunk level, but only for the NDN load balancing decisions.

In contrast to the approaches discussed above, we investigate how to mitigate NDN-specific issues by considering new client-side network measurement approaches specific to NDN combined with generic extensions to adaptation algorithms. While most research relies on changing and upgrading NDN, we focus our effort on dynamic adaptive streaming and tackle the problem by investigating the adaptation algorithm.

TABLE I: Overview of the different parameters in every part of the system.

Zone	Component	Configuration	Instantiations (this work)	Related Work
DASH Mechanisms	Video Content DASH Player	Segment length Adaptation Algorithm	2 seconds BOLA, PANDA, PANDA (extended), AlwaysLowest	[21]
Network Mechanisms	Client Server Cache <i>all</i>	Congestion Control Chunk Size [Number / Capacity / Strategy] Forwarding Strategy	ICP 1400 Bytes {0, 3} / { ∞ , 1500} / LRU ncc	[19, 21] [2] [14, 8, 3]
Network Environment	Containernet	Bandwidth Topology	dynamic[18, 6], static 4×OpenVSwitch, 6×Docker Host	[14]

C. Network Environment

Samain *et al.* [21] introduce the concept of lightweight visualization of NDN hosts using LXC containers. However, it cannot generate flexible topologies. In contrast, a simulation-based evaluation, such as [15, 12, 20] using *ndnSIM* supports flexible topologies as it is based widely on the *ns3* simulator. Yet, it relies purely on simulations, and an emulative or testbed-based verification or results obtained is still necessary.

III. EMULATION ENVIRONMENT

This section provides an overview of our emulation environment used for the evaluations presented in Sections IV to VI, along with the classification presented in Table I.

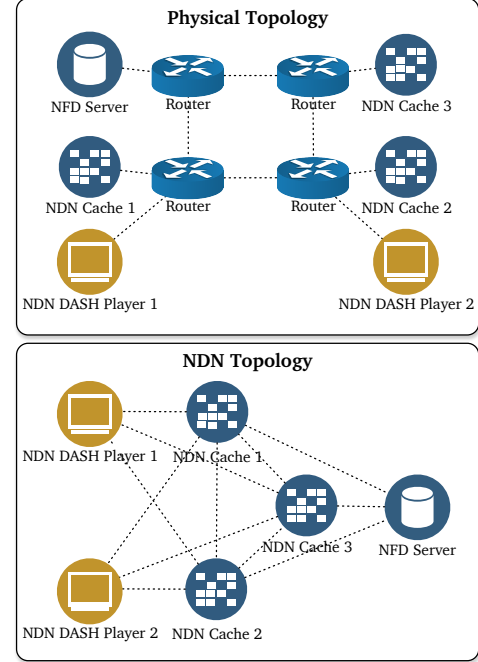
A. Network Environment

Our goal was to evaluate NDN-based DASH, mimicking realistic conditions as closely as possible while maintaining the reproducibility of our results. Thus, our emulation setup needs to fulfill the following functional requirements: i) support for larger, dynamic topologies, ii) within an emulated network environment, iii) based on the original NDN source code. To this end, we built an NDN architecture based on an extension of the *mininet* project called *containernet*¹ that allows to execute Docker hosts as nodes in the mininet emulator.

We claim that more realistic emulation scenarios can be achieved by using *containernet* with encapsulated NDN functionality in Docker containers.

1) *Topology*: The setup used throughout the experiments conducted in this work is depicted in Figure 1. Each of the functional NDN components is represented in separate parts of the network, achieving variations in caching, bandwidth and latency for clients. The architecture varies in its physical setup, which is similar to an ISP topology, and the logical NDN architecture. From a client's perspective, a flat hierarchy between caches exists, and caches can either retrieve content from other caches or from the server.

2) *Bandwidth Traces*: As realistic network conditions are interesting to investigate and confirm the true system performance, we investigate the system using real-world traces collected from mobile devices [18]. We replay these by changing the traffic on the clients' link using *Tc*² accordingly. Whereas these dynamic

**Figure 1:** Evaluation topology using OpenvSwitches.

bandwidth traces allow to analyze the impact in conditions very close to reality, we also use static and designed bandwidth profiles for specific testing scenarios.

B. NDN Mechanisms

In our emulation setup, we distinguish between three types of hosts: *NDN Servers*, *NDN Caches*, and *NDN Clients*. Each type is implemented in a respective container image based on Ubuntu 14.04. All container types run an instance of *NFD*, while caches and the servers use *repo-ng* to host the video dataset.

1) *Chunk Size*: A core parameter of every content-centric network is the size of data that travels with a single chunk. While the default setting is 1000 Bytes, we implemented a chunk size of 1400 Bytes to allow an easy comparison with related work [21].

2) *Congestion Control*: Presently, there is no built-in or standard congestion control for NDN, but research is leading into the direction of adding a congestion control protocol to NDN as it is inevitable in networks that operate at full capacity.

¹<https://github.com/containernet/containernet>

²<http://lartc.org/manpages/tc.txt>

In this work, we use ICP [4, 17], an Interest Control Protocol which realizes a window-based flow control while aiming for efficiency and fairness. Other approaches such as SAID [5] or CCTCP [19] exist but are not directly usable in our NDN infrastructure.

3) *Forwarding Strategy*: Similar to the congestion control, the forwarding strategy in NDN is replaceable. There are many designs and ideas to improve the forwarding in information centric networks for better efficiency and general usability. In the presented studies, we configured the NDN *access* strategy. It supports a faster retransmission of lost interest messages when compared to the (default) *ncc* strategy.

C. DASH Mechanisms

The DASH player deployed on the *NDN Client* is implemented with *libdash*, based on the code provided by [21]. As for the video dataset, a subset of nine layers of the *Big Buck Bunny*³ dataset encoded in two-second H.264-AVC segments is used encoded in bitrates between 1.03 - 4.21 Mbps

IV. CHALLENGES FOR DASH OVER NDN

In the following, we present key challenges for the design of DASH adaptation algorithms for NDN. In particular, we investigate the interaction between NDN's multi-sourcing and caching capabilities and the throughput measurements used in rate-based DASH adaptation algorithms.

To demonstrate the interaction of rate based adaptation algorithms and NDN, we use the following scenario: Given that two clients play the same video in sequential order while using the *AlwaysLowest* adaptation algorithm, i.e., continuously playing the lowest quality, we know that the second client has relevant video segments located in the caches when the playback is initiated. In this setup which mimics Figure 1 the server link bandwidth is set to 4 Mbit/s, and the cache link bandwidths are given by 10 Mbit/s (Cache 1), 20 Mbit/s (Cache 2), 30 Mbit/s (Cache 3). In Figure 2 we show the empirical cumulative density function (CDF) of the measured throughput on NDN chunk basis. All requests from the Client 1 (yellow) are served by the server as the caches are empty. Our first observation here is that although the link bandwidth to the server is fixed to 4 Mbps some chunks are received with a much slower throughput which we attribute to the ICP congestion control in NDN. Note that this throughput measurement, i.e., on NDN chunk basis, is the foundation for the input to rate-based DASH algorithms. For Client 2 about half of the requests are fulfilled with a bandwidth of 10 Mbit/s while the rest is smeared. Again, we observe NDN chunk throughputs at both ends of the scale which we attribute to congestion control. Intermediate bandwidth clusters are caused by ICPs congestion control mechanisms.

Note that on a DASH segment basis, i.e., for aggregating hundreds of chunks into one segment, the throughput measurements are smeared by the effects that single chunks observe. Hence, the existence of caches causes the client to observe

different effective data rates (see Figure 2) which introduces fluctuations between the requests.

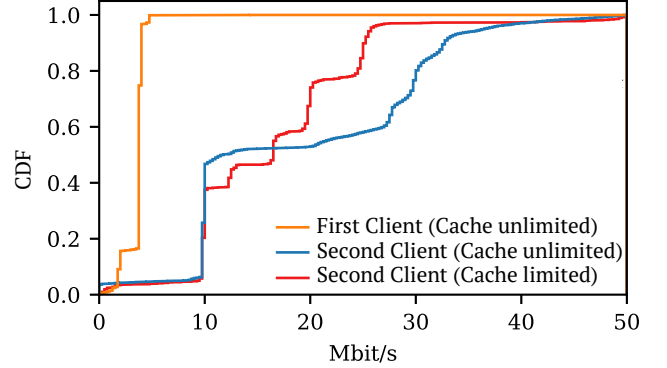


Figure 2: Two clients playing in sequential order using an *AlwaysLowest* adaptation algorithm. The server link bandwidth is given by 4 Mbit/s, while the caches' link bandwidths are 10 Mbit/s (Cache 1), 20 Mbit/s (Cache 2), 30 Mbit/s (Cache 3), respectively. The influence of caching becomes evident by the higher bitrate for the second clients. Cache size for the second client (red) was limited to 5×10^3 NDN chunks.

Overall, we observe that the key point to be addressed for DASH in NDN are the large variations in throughput measurements due to caching which leads to a wrong interpretation of throughput estimates with common adaptation algorithms [14].

V. APPROACH

To address the analyzed challenges, we propose four extensions for throughput-based DASH adaptation algorithms in NDN based on the example of PANDA (see Table II). However, as the discussed extensions are generic, they can also be used with other throughput-based adaptation approaches.

A. Chunk-based Measurements (Extension 1)

The design of NDN allows streaming applications to obtain additional information about NDN's chunk downloads, e.g., the chunk based throughput. For standard DASH over TCP/IP the client receives segment based or byte range based throughput estimates of the connection to the server. In contrast, in NDN, estimates can be obtained for both, DASH segments and (in a finer granularity) NDN chunks. Hence, standard DASH adaptation algorithms may be adjusted or even new adaptation algorithms derived to take advantage of this information.

Since the data packets in NDNs are usually small (a few kilobytes) compared to the video segments (up to many megabytes), the number of measurements for each requested segment is very large. Single NDN chunks are—because of their size—prone to variations when being transmitted through the network as even small variations in transmission time can have a high impact on the throughput estimates. This implies that a high measurement accuracy is required which is also important for increasing the robustness of the feedback to the adaptation algorithm. For example, the correct assignment of Interests being sent off and their matching Data packets coming in at the client is essential to gather the correct timing values

³http://www-itec.uni-klu.ac.at/ftp/datasets/mmsys12/BigBuckBunny/bunny_2s/bunny_2s_1200kbit/

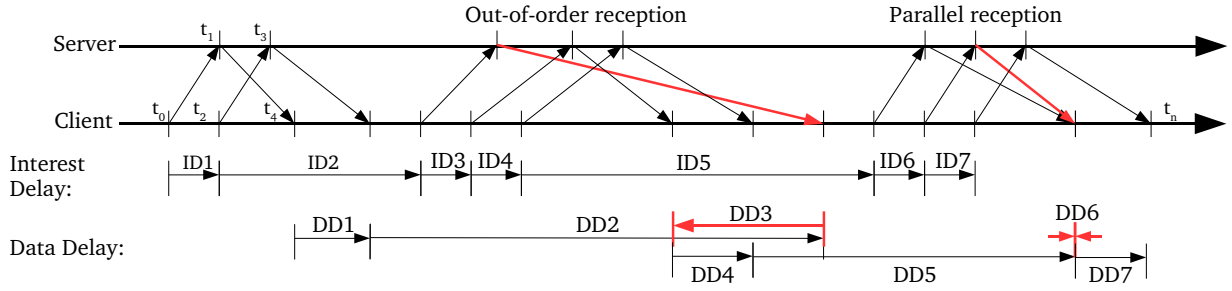


Figure 3: The three different scenarios for Data packet receptions.

for throughput estimations in rate-based or hybrid adaptation algorithms. Other considerations include the pipelining of Interests (sending several Interests in close succession) and possible out-of-order reception of Data packets.

For the throughput measurement based on NDN chunks, we use an approach that has been used for measurements of TCP/IP connection throughput introduced by Khangura *et al.* [11]. The authors suggest a bandwidth estimation from passive measurements based on the *ACK-GAP* model. Note that the TCP sender-side model with ACKs is the inversion of the NDN client-side model: In NDN, the client handles the content flow by sending Interest packets to the server, and the server answers with the sending of Data packets. Lastly, the Data packet is received by the client, and the rate estimation is done. This is the same scenario—but inversed—as in TCP, where the server starts with the data transfer and the client is answering with an ACK.

Since the measurement is conducted over the entire downloading session, no extra probing packets are needed; feedback is extracted constantly with data transfer. However, post-processing of the bandwidth estimations is needed in all cases. Because the measurements are based on the delay of received Data packets that were requested consecutively, we can only estimate the throughput from the second received chunk on, but since the number of feedbacks for the chunk-based granularity is large, i.e., about 2-4 orders of magnitude higher compared to the segment-based granularity in traditional networks, this loss is negligible. Additionally, not all Data packet receptions provide usable timing values when special scenarios occur. The three different scenarios of Data packet receptions are depicted in Figure 3. In the common case, the Data packets are received in the same order as the Interests were sent off and every data delay provides a valid timing value. When an out-of-order reception happens, the delay between two received Data packets is negative ($DD_3 < 0$). The third case is the parallel reception of two Data packets. Here, the data delay equals zero ($DD_6 = 0$). Out-of-order and parallel receptions lead to invalid computation outputs because a not strictly positive timing value is not usable for rate estimations. Thus, the computations based on these two failure scenarios are discarded.

Based on our preliminary investigation, we found that a sample should contain about 50 fine granular computations for providing a very frequent—but reliable—feedback. While a higher sample size up to 100 is still reasonable, samples

using less than 50 computation values suffer more from single measurement failures. It is important to note that the sample size corresponds to the chunk size in NDNs: when larger NDN chunks are used, the measurement failures become less likely. Currently, we use the harmonic mean as a method for sampling.

By considering feedback using more fine granular measurements, we improve the information gain for adaptation algorithms. We argue that in more complex NDN scenarios (e.g., involving multiple caching entities), it is important to get the direct network feedback instead of relying on indirect feedback (as in DASH buffer-based measurements). The main advantage we advocate is a faster detection of source changes within single requests of DASH segments that can be used within the decision-making process of adaptation algorithms.

B. Extended MPD (Extension 2)

While the above method of throughput measurement is an NDN-specific approach, a further general approach to provide a better quality of experience (QoE) is the usage of more precise information before determining an adaptation.

In previous work [10, 25] the authors propose to include the segment size in the MPD.

C. Adaptation Hysteria Reduction (Extension 3)

To address increased adaptations due to throughput fluctuations, we propose two extensions for throughput based adaptation algorithm in order to utilize the advantages of the more precise network measurements as well as the segment size knowledge.

An adaptation algorithm is called unstable, when the video quality fluctuates heavily during playback. To quantify the instability, we employ a measure by [9, 13]. Let r_t be the mean-based quality layer bitrate fetched at time t . The instability for this player at time t is defined in the works above as:

$$I_t = \frac{\sum_{d=0}^{k-1} |r_{t-d} - r_{t-d-1}| \cdot w(d)}{\sum_{d=0}^{k-1} r_{t-d} \cdot w(d)} \quad (1)$$

using $w(d) = k - d$, where w is the weighting function and k is the number of the segments we want to look at in the past. This instability formulation puts a stronger weight on recent adaptations to detect fluctuations, hence, $I_t \geq 0$. For no fluctuations, I_t is zero. We use $k = 10$ to increase the weight of more recent segments without looking too far into past adaptation decisions. In subsequent adaptation decisions, if

$I_t > 0.05$ adaptation to higher qualities is postponed to reduce short-term fluctuations.

D. Local Segment Rate Variation (Extension 4)

To address the rate sustainability of adaptation decisions we use the following algorithm that we denoted as local segment rate variation function. This time, we use R_t to denote the segment-level quality bitrate of the specific video segment at time t (see Section V-B) and B_t , which is the currently measured bandwidth, as described in Section V-A. For each triggered adaptation to a higher quality level, the following decision process is executed: If *skipAdaptation* is set to `true`,

Algorithm 1: Process to determine if a planned bitrate change can be supported for the next i segments

```

window: 3;
skipAdaptation: False;
for  $1 \leq i \leq \text{window}$  do
    if  $B_t \leq R_{t+i}$  then
        skipAdaptation = True;

```

a planned adaptation to higher quality is skipped. This extension is crucially different from standard rate adaptation comparisons as it takes into account the *local variation* of the segment sizes. Therefore, adaptations to higher quality are only permitted if they can be provided over a longer period of time and not just for a single video segment. After testing different values, we recommend a window size of 3 segments because a bigger window would impede the adaptation algorithm too much while a smaller window would not stabilize the video quality good enough. It is important to know that this value is dependent on the segment length. Therefore, in the used configuration, a windows size of 3 means that we want to sustain the current quality at least for 3 segments which equals to 6 seconds of playback time.

TABLE II: Comparison between the proposed extensions to PANDA represented by adaptation and video quality metrics. The used test-cases did not exhibit any stalling.

Algorithm Used extensions	PANDA 1-2	PANDA 1-3	PANDA 1-4
Adaptations [#]	80	58	29
Magnitude [level]	0.63	0.38	0.18
Mean-based Bitrate [Mbit/s]	2.66	2.85	2.42
Segment-level Bitrate [Mbit/s]	2.31	2.57	2.20

Table II shows results for the chunk-based measurement method on the example of the PANDA algorithm by comparison with the instability reduction, local segment rate variation, and extended MPD approaches. Overall, only the third PANDA configuration can provide a switching magnitude that is low enough for an argumentatively good QoE while still providing an overall good bitrate. When PANDA is extended with the instability function exclusively, there are still too many switches which last only for one single segment. This harms the QoE more than the higher bitrate can improve the QoE.

VI. EVALUATION

TABLE III: Overall comparison of adaptation, video quality, and stalling metrics.

	BOLA		PANDA		PANDA (all extensions)	
Client	C_1	C_2	C_1	C_2	C_1	C_2
Adaptations [#]	19	18	80	79	20	26
Magnitude [level]	0.2	0.2	0.8	0.8	0.2	0.3
Stalling sum [s]	2	4	7	13	0	0
Stalling [#]	2	4	5	6	0	0
Bitrate [Mbit/s]	2.3	2.1	2.0	2.2	2.3	2.5

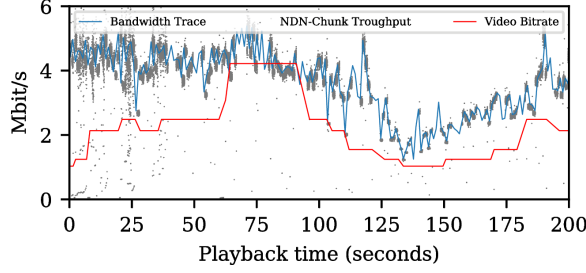
To evaluate the performance of the discussed DASH adaptation algorithm we are using a trace-based emulation of the client bandwidth as described in Section III. The server link capacity is 4 Mbit/s, and the Cache link capacities are given by 10 Mbit/s (Cache 1), 20 Mbit/s (Cache 2), and 30 Mbit/s (Cache 3), respectively. Playback is initiated sequentially for two clients, i.e., the second client encounters a situation where content is already cached. We test the three adaptation algorithms discussed separately, i.e., *BOLA*, *PANDA* and the *extended version of PANDA* that is proposed in this work. To ensure a fair comparison between these algorithms, a maximum playout buffer size of 20 seconds is configured in each case.

Investigating the respective first clients in Table III we observe that stalling time is overall shorter compared to the second clients indicating to the prior discussed effect of inadequate bandwidth estimation once content has been cached. This effect is particularly pronounced in the regular version of PANDA. The extended version of PANDA, as proposed in our work, mitigates this effect resulting in no stalling for both, the first and second client.

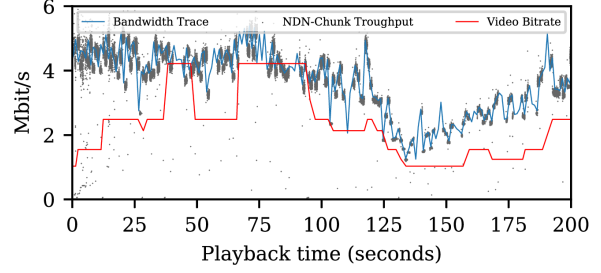
While both PANDA-based adaptation algorithms achieve an increase in the playback bitrate compared to the first client, this is not the case for the buffer-based adaptation algorithm BOLA. Here, in case of the second client, the higher available bandwidth, available by caches populated with previously requested segments, is not sufficiently utilized. This results in a lower average bitrate for the second client.

Overall, the extended version of PANDA shows a stable playback behavior, which is comparable to BOLA in terms of adaptation steps for the first clients. While in the extended PANDA adaptation algorithm's case, the second client shows more adaptation steps compared to BOLA, it maintains a high bitrate without stalling. Thus, it can be argued that the negative effect of a slightly increased number of adaption steps is counterbalanced by the higher bitrate and no stalling. We expect that the fact that no stallings occur at a higher bitrate leads to a higher QoE that can be achieved within a multi-cache NDN scenario when using our extended version of PANDA.

The NDN chunk throughput shown in Figure 4, for the extended version of PANDA, indicates that obtained chunk-based bandwidth samples follow the bandwidth trace for most measurements. This observation is also confirmed when



(a) Client 1 extended PANDA.



(b) Client 2 extended PANDA.

Figure 4: Extended PANDA clients playing in sequential order, i.e., (a) followed by (b), applying the same bandwidth trace for both clients.

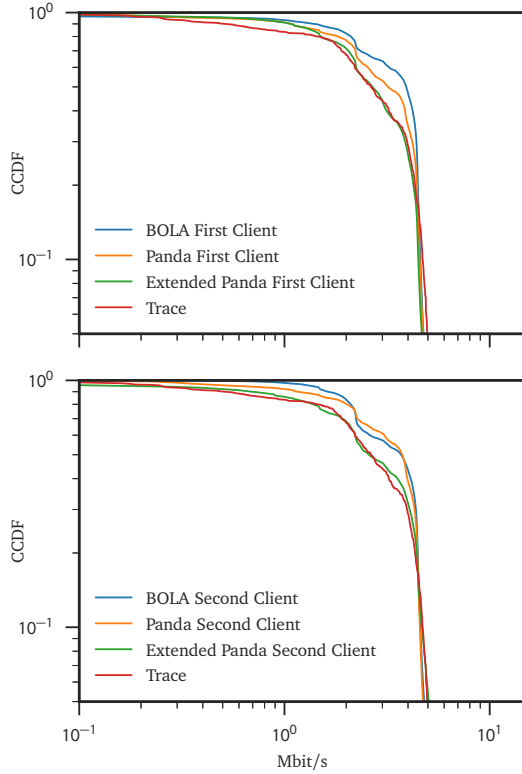


Figure 5: Comparison of the investigated adaptation algorithms: BOLA, PANDA, and extended PANDA. For each, two clients play in sequential order applying the same bandwidth trace for each client. The cache size is limited to 1500 Segments. Server bandwidth was 4 Mbit/s, Caches bandwidth 10 Mbit/s, 20 Mbit/s, 30 Mbit/s respectively.

compared with the distribution of these measurements depicted in Figure 5, where roughly 80% of the throughput traces' distribution is closely resembled by measurements obtained in the extended version of PANDA.

When comparing these distributions between all clients and adaptation algorithms, the underlying reasons for an improvement of the playback bitrate of the proposed extended PANDA become evident; while all algorithms exhibit some degree of cache usage (segments may be also fetched from caches due to retransmissions in case of the first clients) we observe a comparatively large share of segments from

bandwidths clustering around the network capacity of caches for the second client in case of the extended version of PANDA. Therefore, overall a higher bandwidth for segments requests can be achieved beyond the bottleneck capacity of the server. Last, we also observe that roughly 1% of all throughput measurements are still overestimated, due to burstiness in traffic patterns introduced by switches and the ICP congestion control (this is omitted in Figure 5).

VII. CONCLUSIONS AND FUTURE WORK

This paper presents a systematic study on the effect of NDN in DASH and presents evidence for limitations of current DASH adaptation algorithms when used in this context based on emulative evidence. The root causes are variations in the traffic measurements, which underlie all standard DASH adaptations algorithms. These variations are more critical in NDNs than in traditional network setups: in an NDN each request can be served by any suitable NDN repository, DASH clients are not aware of it. Thus, the resulting variation in bandwidth from a client's perspective leads to sub-optimal adaptations decisions. Hence, new approaches for DASH adaptation algorithms considering the specific NDN context are necessary. We have thus introduced new concepts for more fine-granular and precise measurements based on NDN chunks instead of DASH segments, as well as the use of additional information about the chunk sizes. By doing so, we demonstrate an improvement in the QoE over the existing adaptation algorithm PANDA. In general, the proposed extension can also be used for other throughput-based adaptation algorithms in NDNs. While the contributions in this work are demonstrated on the example of NDN, the extensions to the PANDA adaptation algorithm regarding a reduction of adaptations steps and improved segment size awareness are general, i.e., they may also be deployed on the context of TCP-based DASH.

Last, we found that the observed performance is strongly related to specific combinations of configurations, including the cache size and the used congestion control algorithm in NDNs. Such cross-layer network mechanisms should be considered as part of future studies.

ACKNOWLEDGMENT

This work has been funded by the German Research Foundation (DFG) as part of the project C03 in the Collaborative Research Center (SFB) 1053 MAKI.

VIII. REFERENCES

- [1] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, "A survey of information-centric networking," *IEEE Communications Magazine*, vol. 50, no. 7, 2012.
- [2] S. Awiphan, T. Muto, Y. Wang, Z. Su, and J. Katto, "Video streaming over content centric networking: Experimental studies on PlanetLab," *Computing, Communications and IT Applications Conference (ComComAp)*, pp. 19–24, 2013.
- [3] D. Bhat, C. Wang, A. Rizk, and M. Zink, "A load balancing approach for adaptive bitrate streaming in Information Centric networks," *IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, 2015.
- [4] G. Carofiglio, M. Gallo, and L. Muscariello, "Icp: Design and evaluation of an interest control protocol for content-centric networking," in *2012 Proceedings IEEE INFOCOM Workshops*, 03/2012, pp. 304–309.
- [5] J. Chen, M. Arumathurai, X. Fu, and K. Ramakrishnan, "Said: A control protocol for scalable and adaptive information dissemination in icn," in *Proceedings of the 3rd ACM Conference on Information-Centric Networking*, ACM, 2016, pp. 11–20.
- [6] (09/2017). Dash industry forum: Dash-avc/264 test cases and vectors., [Online]. Available: <http://dashif.org/wp-content/uploads/2015/04/DASH-AVC-264-Test-Vectors-v09-CommunityReview.pdf>.
- [7] B. Feng, H. Zhou, and Q. Xu, "Mobility support in named data networking: A survey," *EURASIP Journal on Wireless Communications and Networking*, vol. 2016, no. 1, p. 220, 09/15/2016.
- [8] R. Grandl, K. Su, and C. Westphal, "On the interaction of adaptive video streaming with content-centric networking," *20th International Packet Video Workshop (PV)*, 2013.
- [9] J. Jiang, V. Sekar, and H. Zhang, "Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive," *IEEE/ACM Transactions on Networking (TON)*, vol. 22, no. 1, pp. 326–340, 2014.
- [10] P. Juluri, V. Tamarapalli, and D. Medhi, "Sara: Segment aware rate adaptation algorithm for dynamic adaptive streaming over http," *IEEE International Conference on Communication Workshop (ICCW)*, pp. 1765–1770, 2015.
- [11] S. K. Khangura and M. Fidler, "Available bandwidth estimation from passive tcp measurements using the probe gap model," in *Proceedings of IFIP Networking*, 06/2017.
- [12] W. Li, S. M. A. Oteafy, and H. S. Hassanein, "On the performance of adaptive video caching over information-centric networks," in *2017 IEEE International Conference on Communications (ICC)*, 05/2017, pp. 1–6.
- [13] Z. Li, X. Zhu, J. Gahm, R. Pan, H. Hu, A. C. Begen, and D. Oran, "Probe and adapt: Rate adaptation for http video streaming at scale," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 4, pp. 719–733, 04/2014.
- [14] Y. Liu, J. Geurts, J. C. Point, S. Lederer, B. Rainer, C. Muller, C. Timmerer, and H. Hellwagner, "Dynamic adaptive streaming over CCN: A caching and overhead analysis," in *IEEE International Conference on Communications (ICC)*, pp. 3629–3633.
- [15] B. Rainer, D. Posch, and H. Hellwagner, "Investigating the Performance of Pull-Based Dynamic Adaptive Streaming in NDN," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 8, pp. 2130–2140, 08/2016.
- [16] A. Ramakrishnan, C. Westphal, and J. Saltarin, "Adaptive video streaming over CCN with network coding for seamless mobility," in *IEEE International Symposium on Multimedia (ISM)*, 12/2017, pp. 238–242.
- [17] Y. Ren, J. Li, S. Shi, L. Li, and X. Chang, "An interest control protocol for named data networking based on explicit feedback," in *Proceedings of the Eleventh ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS)*, Oakland, California, USA, 2015, pp. 199–200.
- [18] H. Riiser, P. Vigmostad, C. Griwodz, and P. Halvorsen, "Commute path bandwidth traces from 3G networks," in *Proceedings of the 4th ACM Multimedia Systems Conference (MMSys)*, New York, New York, USA: ACM Press, 02/2013, pp. 114–118.
- [19] L. Saino, C. Cocora, and G. Pavlou, "Cctcp: A scalable receiver-driven congestion control protocol for content centric networking," in *IEEE International Conference on Communications (ICC)*, 06/2013, pp. 3775–3780.
- [20] J. Saltarin, E. Bourtsoulatz, N. Thomos, and T. Braun, "Adaptive video streaming with network coding enabled named data networking," *IEEE Transactions on Multimedia*, vol. 19, no. 10, pp. 2182–2196, 2017.
- [21] J. Samain, G. Carofiglio, L. Muscariello, M. Papalini, M. Sardara, M. Tortelli, and D. Rossi, "Dynamic Adaptive Video Streaming: Towards a Systematic Comparison of ICN and TCP/IP," *IEEE Transactions on Multimedia*, vol. 19, no. 10, pp. 2166–2181, 10/2017.
- [22] K. Spiteri, R. Uргаonkar, and R. K. Sitaraman, "BOLA: Near-optimal bitrate adaptation for online videos," in *IEEE INFOCOM*, 2016.
- [23] D. Stohr, F. Beji, R. Dwarakanath, R. Steinmetz, and W. Effelsberg, "Mobile NDN-Based Dynamic Adaptive Streaming over HTTP," in *IEEE Conference on Local Computer Networks (LCN), Demo*, 2016, p. 3.
- [24] D. Stohr, A. Frömmgen, A. Rizk, M. Zink, R. Steinmetz, and W. Effelsberg, "Where are the sweet spots?: A systematic approach to reproducible dash player comparisons," in *Proceedings of ACM Multimedia*, 2017, pp. 1113–1121.
- [25] C. Wang, A. Rizk, and M. Zink, "SQUAD: a spectrum-based quality adaptation for dynamic adaptive streaming over HTTP," *Proceedings of the 7th International Conference on Multimedia Systems (MMSys)*, 1:1–1:12,
- [26] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, K. Claffy, P. Crowley, C. Papadopoulos, L. Wang, and B. Zhang, "Named data networking," *ACM SIGCOMM*, vol. 44, no. 3, pp. 66–73, 07/2014.