Dynamic Adaptive Video Streaming: Towards a systematic comparison of ICN and TCP/IP

Jacques Samain [*†], Giovanna Carofiglio [†], Luca Muscariello [†], Michele Papalini[†], Mauro Sardara[†], Michele Tortelli[*] and Dario Rossi[*]
[*] Telecom ParisTech, Paris, France – first.last@telecom-paristech.fr
[†] Cisco Systems, Issy les Molineaux, France – first.last@cisco.com

***Abstract*—Streaming of video content over the Internet, both to and from mobile devices, is experiencing an unprecedented growth, with emerging formats such as 4K and beyond. While video permeates every applications, it also puts tremendous pressure in the network – to support users having heterogeneous accesses and expecting high quality of experience, in a furthermore cost-effective manner. It this context, Future Internet (FI) paradigms such as Information Centric Networks (ICN) are particularly well suited to not only enhance video delivery at the client (as in the current DASH approach), but to also naturally and seamlessly extend video support deeper in the network functions.**

In this paper, we contrast ICN and TCP/IP with an experimental approach, where we employ several state-of-the art DASH controllers (PANDA, AdapTech, and BOLA) on an ICN vs a TCP/IP network stacks. Our campaign, based on tools which we developed and make available as open-source software, includes multiple videos (up to 4K resolution), channels (e.g., DASH profiles, emulated WiFi and LTE), and levels of integration with an ICN network (i.e., vanilla NDN; wireless loss detection and recovery at the access point; load balancing). Our results clearly illustrate, as well as quantitatively assess, sizable benefits of the ICN paradigm for video streaming, as well as warns about potential pitfalls that are however easy to avoid.

I. Introduction

There is no doubt about video, and especially mobile video predominance in future traffic trends. According to Cisco VNI forecast, more than 80% of all IP traffic will be video and two-third of all Internet traffic will be generated from wireless and mobile devices by 2020 [1]. Traffic growth goes hand in hand with evolving video services (e.g., UHD 4K-8K video, Virtual/Augmented Reality), driving future 5G networks design to meet new mobile video usages with very-high bandwidth requirements under ultra-low latency constraints. Also, a significant change in video consumption has been observed, with a clear transition from traditional multi-channel broadcast TV to adaptive streaming over an increasingly heterogeneous and mobile network access.

All these factors put pressure on the capabilities of future 5G networks and highlight their critical role in the support of Dynamic Adaptive Streaming (DAS). With DAS, we refer here to the variety of techniques, in most of the cases relying on HTTP, that have bloomed in the last years to realize an efficient multimedia delivery over the Internet: many popular ones are proprietary (e.g., Apple HLS, Microsoft HSS), while Dynamic Adaptive Streaming over HTTP (DASH) has recently become a standard. Since DAS techniques were initially designed for CDN/OTT content delivery, their interaction with the network has been only superficially studied so far. In the 5G mobile and heterogeneous network access, it seems of utmost importance to consider DAS interaction with the network and to move caching and computing capabilities to the network edge in order to enable efficient mobile video delivery [2] . Given this context, Information-Centric Networking (ICN) [3] appears as a natural network substrate for DAS [4]–[12].

ICN proposes a content-centric communication paradigm that leverages location-independent network names and a content-aware connectionless transport including network-level caching, multi-path forwarding capabilities and seamless mobility support – features that are all very appealing for DAS systems. However, the potential for ICN application in adaptive streaming services as an alternative to relieve from some of the recognized inefficiencies of standard TCP/IP transport has been only partially explored (refer to [13] for an overview of ICN aspects related to video delivery). Recently, valuable work started to appear [4]–[12], which gives hints on the potential benefits coming by exploiting capabilities of an ICN content-aware network (namely, built-in caching and name-based forwarding) to assist DAS rate adaptation inside the network, rather than only at the client side. At the same time, the literature currently lacks a systematic approach for testing the interplay of ICN and DAS. Similarly, a quantification of the benefits ICN could bring over the current TCP/IP solutions in realistic environments is far from being complete. In this paper, we take a step back and:

- we review existing state-of-the-art DAS rate adaptation strategies, and select three that are representative of the whole design space;
- we develop a platform for experimental evaluation of these DAS strategies over both ICN and TCP/IP in realistic wired/wireless environments, that we make available as open-source software [14];
- we carry on an experimental campaign of DAS over ICN vs TCP/IP, systematically assessing benefits (or warning about potential pitfalls) coming from ICN building blocks such as enhanced rate adaptation, in-network loss recovery, or load balance among heterogeneous interfaces.

In the remainder of the paper we first state more clearly the problem and our objectives (Sec.II), and we overview the DAS literature over TCP/IP and ICN, to select those controllers we use throughout the paper (Sec.III). We next introduce the

architecture, the emulation platform and scenarios (Sec.IV) over which our experimental results are gathered (Sec.V-Sec.VI). Finally, we summarize the main lessons learned (Sec.VII).

II. Problem Statement

Information Centric Networking (ICN) architectures, with their content-centric view of Internet communications and distinctive features like pull-based approach, in-network caching, natural support for mobility, multi-cast, and multi-path communications [3], seem to perfectly fit in the design space of client-pull video streaming systems. It is not by chance that recent literature [4]–[12] considers ICN a valuable alternative to TCP/IP for improving the efficiency of current video streaming systems (see Sec.III). This section briefly reviews ICN characteristics at the light of DAS requirements, and discuss the potential advantages in adopting ICN for DAS video delivery.

ICN at a glance. Among the numerous ICN architectures, we focus our attention here to the one currently under discussion at Information-Centric Networking Research Group (ICNRG) [15] and intended to unify Named Data Networking (NDN) [16] and Content-Centric Networking (CCN) [17]. NDN and CCN are two prominent and very similar ICN architectures whose differences do not affect the description and the considerations of this paper. However, since our experimental campaign is based on the NDN Forwarding Daemon [18] and to avoid ambiguities in the following, we refer to the reference architecture as NDN in the following. In NDN, content chunks are identified by unique names, requested by the user via *Interests* packets and retrieved as *Data* packets with the same name. To enable symmetric routing of Data towards the requesting users, NDN routers keep track of ongoing requests in a *Pending Interest Table* (PIT) of the faces Interests originate from. NDN routers also have the capability to locally store Data, in what is called *Content Store* (CS): if a matching Data packet is found into the CS, it is delivered using the state information from the corresponding PIT entry. Otherwise, in case of cache miss, a Longest Prefix Match of the content name is looked for into the node's *Forwarding Information Base* (FIB). The FIB, populated by a name-based routing protocol, provides one or multiple egress faces per routable name prefix. Then, the Interest packet gets forwarded according to a configured strategy, e.g., Shortest Path, Broadcast, Load Balancing (LB).

Connection-less pull-based transport. NDN leverages a pull-based transport, where rate and congestion are controlled by the receiver, similarly to DAS, where rate adaptation is decided at the client. Interests are forwarded by name in a dynamic hop-by-hop fashion by traversed routers, and, once satisfied, the matching Data packet is sent over the reverse path. As a result of the addressing-by-name principle, NDN transport model overcomes the static binding between an object and a location identifier: the receiver issues name-based packet requests over possibly multiple network interfaces with *no connection instantiation* and *no a priori knowledge of the content source* (hitting cache or repository). As a consequence, NDN simplifies mobility/connectivity disruption management, not requiring any connection state migration in case of end-user mobility or path failure.

Unlike in the TCP/IP world, there is not currently a default transport protocol for NDN, for which we leverage our previous work about Interest Control Protocol (ICP) [19]. Robustness to mobility/path disruption and efficient multi-source delivery via receiver-controlled multi-path are results of the agility of the ICP transport, especially in a mobile dynamic and heterogeneous network environment where the early connection binding and the sender-based nature of TCP have proved to introduce inefficiencies [20]. At the same time, we show that it is necessary to compensate for missing features of TCP/IP in ICP (e.g., end-to-end loss recovery) at NDN network-level.

In-network control. Soft-state associated to pending requests enables fully distributed in-network decisions that may help rate, loss, mobility, and congestion control management, otherwise performed at the consumer side only. Also, the content-awareness provided by names to network nodes enables a different use of buffers, not only to absorb input/output rate unbalance, but for temporary caching of in-transit Data packets, *reuse* (subsequent requests for the same Data can be served locally with no need to fetch data from the original server/repository, useful in a multicast scenario) and *repair* (packet losses can be recovered in the network, with no need for the sender to identify and retransmit the lost packet, useful in a wireless access).

Aspects of in-network control can then significantly improve an ICP/NDN DAS over the current TCP/IP DASH under certain aspects, like hop-by-hop rate and congestion control [21], in-network loss detection and recovery [22], joint forwarding-caching strategies [23, 24], and multicast capabilities [25]. In this paper we study the most useful among these NDN building blocks, illustrating and quantifying the benefits that they bring over TCP/IP, as well as potential downsides their careless use might introduce.

III. Background

Most of the DAS literature has with few exceptions [34, 35] focused on *application-level and client-side* adaptation of the requested video quality [26]–[33], [42]. More recently, work started to appear that additionally considered *in-network functionalities* offered by an ICN paradigm to support DAS [4]–[12].

A summary of the most relevant work in the literature is provided in Tab. I, clearly separating work in the TCP/IP (top) vs ICN (bottom) domains. Following a consolidated taxonomy [36], DAS strategies can be classified into one of two big families: **rate-based (RB)** or **buffer-based (BB)**,

TABLE I
STATE OF THE ART IN DYNAMIC ADAPTIVE VIDEO STREAMING. (ITEMS HIGHLIGHTED IN BOLD ARE USED IN THE EXPERIMENTAL CAMPAIGN)

	Reference	Tool	Main Approach	Buffer Level	Avg Bitrate	Quality Switches	Rebuffering	Start-up Latency	Throughput	Delay	Fairness
TCP/IP	FESTIVE [26]	*Experiments*	*RB*	C	O	O			C		O
	PANDA [27]	***Experiments***	***RB***	**C**	**O**	**O**	**O**		**C**		**O**
	BOLA [28]	***Experiments***	***BB***	**C**	**C**	**O**	**O**				
	AdapTech [29]	***Experiments***	***BB***	**M**	**M**				**M**		
	ELASTIC [30]	*Experiments*	*BB*	C	O	O	O		C		O
	BBA-x [31]	*Experiments*	*BB*	C	O	O	O		C		
	Miller('12) [32]	*Experiments*	*BB*	C	O	O	O		C		
	BIEB [33]	*Heuristic*	*BB*	C	C/O	O	O	O	O		
	Essaili('13) [34]	*Simulation*	*INA*		M						
	QFF [35]	*Optimization*	*INA*			O			C		O
	Thang('14) [36]	*Experiments*	*Investigation*	M	M				M		
	Huang('12) [37]	*Experiments*	*Investigation*		M				M		
	Thang('12) [38]	*Experiments*	*Investigation*		M				M		
	Akhshabi('13) [39]	*Experiments*	*Investigation*		M	M			M		
	Dobrian('11) [40]	*Conviva*	*Measurements*		M		M	M			
	YouSlow [41]	*Chrome*	*Measurements*		M	M	M	M			
	xMPC [42]	*Optimization*	*BB/RB*	C	C	O	O	C	C		
	LCC [43]	*Optimization*	*Offline*						O	C/O	
ICN	Lederer('14) [4]	*Emulation*	*Investigation*	M					M		
	Lederer('13) [5]	*Emulation*	*Investigation*	M	M	M			M		
	DASC [6]	*Simulation*	*Investigation*		O				C		
	Petrangeli('15) [7]	*Simulation*	*Investigation*	M	M	M					
	DASH-INC [8]	*Model*	*Characterization*		M						
	Bath('15) [9]	*Experiments*	*INA*							M	
	INA [10]	*Simulation*	*INA+BB*						C		O
	DASCache [11]	*Optimization*	*Offline*						O	C	
	Rainer('16) [12]	*Simulation*	*Investigation*		O				C		

Legend: O: objective metric; C: control metric; M: measured metric.
BB: buffer-based; RB: rate-based; INA: in-network adaptation.

meaning that the adaptation is performed mainly[1] by considering either the *estimated throughput* or the *buffer level*, respectively (denoted as "main approach" in Tab. I). The table additionally reports, for each work, the tools adopted to design the proposed DAS strategy (or to carry on the proposed analysis), and a set of Key Performance Indicators (KPIs) (including throughput, buffer level, quality switches, rebuffering events, startup latency, fairness, etc.) used as either **control (C)** knob, **objective (O)** of the algorithm, or **measured (M)** metric. In what follows, we briefly overview the full landscape but, for reason of space, provide more details of few strategies that we select as representative of each class, namely Probe AND Adapt (PANDA) [27] (mostly RB), Buffer Occupancy based Lyapunov Algorithm (BOLA) [28] (mostly BB), and AdapTech [29] (equal balance between BB and RB).

[1]Despite this coarse distinction, in all the surveyed strategies, both metrics (i.e., throughput and buffer level) are often jointly considered in order to obtain a finer adaptation. However, according to the importance that each metric has in the whole decisional process, it is still possible to classify the strategy of interest as either *mainly RB* or *mainly BB*.

A. Rate-based strategies (TCP/IP)

The general idea of Rate-based (RB) algorithms [26, 27] is that of using the measured throughput of the last segment, $\widetilde{C}_k$, as an *estimate* for the throughput of the next segment $\widehat{C}_{k+1}$. In turn, this knowledge assists the selection of the highest affordable quality (i.e., $rate(q_{k+1}) < \widehat{C}_{k+1}$) to be requested. Pure RB algorithms, however, suffer from inefficiencies [37] like: *rebufferings*, bandwidth *underutilization* (linked to the so-called downward-spiral effect) or *overestimation* (due to the ON-OFF pattern generated by the interaction with TCP congestion control), *instability* (i.e., fluctuating estimates caused by short-term variations of the bandwidth), and *unfairness* (some client might be forced to request a lower quality w.r.t. their fair share). Several proposals exist to address the aforementioned issues at client [27, 26], server [39], and/or network [34, 35] viewpoints.

PANDA. The strategy proposed in [27], namely Probe and Adapt (PANDA), takes inspiration from TCP congestion control, implementing the same principles at the application layer (i.e., operating at a video-segment rather than at RTT

timescale). The main observation is that throughput estimates are accurate (i.e., they reflect the fair-share bandwidth) only when links are oversubscribed and with no OFF intervals (i.e., when clients are idle). In the remaining cases, overestimations occur. The idea is then to constantly *probe* the available bandwidth by varying the requested bitrate. Since bitrates associated to available video qualities are discrete, intervals between consecutive requests for video segments are fine-tuned in order to obtain a *continuous* average data rate sent over the network: the average data rate is used to probe the bandwidth until congestion (i.e., network conditions cannot sustain the requested bitrate, and a back off should occur), and determine inter request time.

In a nutshell, PANDA comprises four main steps: (i) Additive Increase Multiplicative Decrease (AIMD)-like *bandwidth estimation*, to compute a target average data rate; (ii) Exponential Weighted Moving Average (EWMA) *smoothing* of the previous target rate; (iii) *quantization* of the smoothed estimate in order to compute the quality to be requested (it is a dead-zone quantizer with upshift, Δ_{up}, and downshift, Δ_{down}, safety margins which mitigate frequent bitrate shifts between two adjacent levels); (iv) *scheduling* of the next segment request to comply with a target inter-request time (i.e., if the actual download time is smaller than this target, the client will wait a time equal to their difference in order to download the next segment). Compared to others conventional rate-based players, PANDA is shown to have the best stability-responsiveness tradeoff, for which we select it as representative DAS strategy.

B. Buffer-based strategies (TCP/IP)

The general idea of Buffer-based (BB) algorithms [28]–[33], instead, is to select the video quality according to the current buffer occupancy $B(t)$. Typically, the buffer is divided into multiple ranges, and different actions are taken according to its actual level. A general policy is that of requesting the lowest quality when the buffer is nearly empty, or below a minimum threshold, B_{min}, in order to avoid rebufferings; conversely, the highest quality can be requested when the buffer is above a maximum threshold B_{max}. To handle the remaining cases (i.e., $B_{min} \leq B \leq B_{max}$), a proper function (e.g., monotonically increasing) is needed to map any possible combination between buffer occupancy and requested video quality inside the feasible region. Segments that accumulate into the buffer can act as a cushion to absorb the effects of small bandwidth variations; however, if the mapping spacing between two consecutive bitrates is too narrow (e.g., number of available qualities too high compared to the buffer range), unwanted quality switches could arise.

BOLA. Bitrate adaptation is tackled as a utility maximization problem by BOLA [28]. The goal it that of designing a control algorithm that maximizes a joint utility $\bar{v}_N + \gamma \bar{s}_N$, where $\bar{v}_N$ is the time-average playback quality computed over the N segments of the video (a logarithmic function is used to compute each single term), $\bar{s}_N$ is the average playback smoothness (i.e., the fraction of time spent not rebuffering), and γ is a weighting parameter which allows to prioritize between the two metrics. Through problem relaxation, the authors conceive an online version of BOLA, where, at each time-slot, adaptation is made by monitoring the current buffer level and by solving a deterministic optimization problem, whose constraints are those of keeping the buffer as much stable as possible, and maximizing the aforementioned utility function. Different variants of the main strategy are also proposed in order to either minimizing the number of quality shifts (i.e., since a bitrate capping is introduced by monitoring the available bandwidth, utility can be sacrificed), or maximizing the utility (with more quality variations). BOLA is the default strategy implemented in the DASH.js player [44].

AdapTech. A stronger coexistence between BB and RB decision processes is realized in AdapTech [29]. The principal aim is that of stabilizing the buffer level around a target value, B_{max}, while keeping the quality as smooth as possible, i.e., not reacting to short term spikes in the available bandwidth, and avoiding rebuffering events. The algorithm requires the use of two thresholds (B_{min} and B_{max}), and two different available bandwidth estimates (throughput of the last segment, A, and its smoothed version, $\hat{A}$, via EWMA). AdapTech is divided into two phases: *Buffering-State* and *Steady-State*. In *Buffering-State*, a segment is downloaded right after the end of the download of the previous one in order to quickly build up the buffer. Once the target value, B_{max}, is reached, AdapTech enters in *Steady-State*, where a new segment is downloaded only after a previous segment is removed from the buffer (i.e., has been played by the video player). The decrement/increment of the requested video quality are governed by two different logics: as for the *decreasing* phase, when the $B(t) > B_{max}$, the algorithm keeps the current quality, to avoid overreaction to negative spikes in the available bandwidth, as the buffer can absorb short-term variations. When the buffer level is between $B_{min} \leq B(t) \leq B_{max}$, the algorithm quickly adapts by switching to a lower sustainable quality (i.e., $rate(q-x) \geq A$). Finally, the lowest quality is always requested when $B(t) < B_{min}$. As for the *increasing* logic, instead, if the buffer level is between $B_{min} \leq B(t) \leq B_{max}$, the current quality q is incremented provided that the requested bitrate is sustainable ($rate(q+1) \geq A$). If the buffer level is higher than B_{max}, then the quality is increased only if two conditions are jointly met: over the last T seconds, the video bitrate at the current quality is smaller than the smoothed estimate $\hat{A}$; in addition, the requested bitrate for the next segment is smaller than the instantaneous bandwidth A. These conditions avoid that positive short-term fluctuations of the bandwidth induce unwanted oscillations of the video quality. We consider AdapTech as representative of the hybrid BB/RB family.

C. Beyond client-based adaptation (ICN)

As previously stated, an adaptive video streaming service might take advantage, at a relatively low cost, from built-in features of ICN, like in-network caching, multi-cast and multi-path support. For this reason, despite some initial work

assessing the performance of rate-based algorithms for Named Data Networking (NDN) [4], most of the literature on video streaming and ICN has proposed and investigated *in network* adaptation mechanisms.

Studies range from the possibility to dynamically select the best performing link (i.e., between 4G and Wifi) when downloading a video segment in a mobile scenario [5] (thus reaching better performance than the classic scenario with a single link), to the usefulness of caching in the presence of multiple clients fetching the same content [6] (thus resulting in an increment of the retrieved video quality over time). The picture is however far from being complete. For instance, some argues [7] that the presence of in-network caching may favor the use of Scalable Video Coding (SVC) for an ICN-based adaptive streaming service, since the layered approach could increase the efficiency and the flexibility of the adaptation process (i.e., as base layers can be prioritized over enhancement layers in order to guarantee a continuous video playout if the latter ones cannot be retrieved). At the same times, others point out that this could induce some inefficiencies, like quality oscillations [8, 9] due to hit/miss events interfering with the bandwidth estimation process, or even client starvation [10] (i.e., consumers of unpopular contents in competing DASH streams might perceive an unacceptable quality w.r.t. actual network conditions).

Possible solutions propose to increase the decisional and computational power of intermediate nodes with, for example, the ability of either altering the media description according to the available bitrates in their caches, or transcoding the cached qualities depending on received requests [8] (i.e., they can act like producers). Another solution, namely In-Network Adaptation (INA) [10], is based on the active participation of intermediate ICN routers, which assesses whether the received requests cannot be satisfied by the network, and send back NACKS as a form on notification. When the in-network adaptation envelope is pushed too far, however scalability issues may be encountered: e.g., this mat be the case of studies such as [11] where an orchestrating entity, aware of the current network state, should be able to optimally place contents inside network caches by solving an Integer Linear Programming (ILP) optimization problem, with the aim of minimizing the average access time per bit.

Differently from previous work, our aim is not to explore how the performance of a specific DAS algorithm, in furthermore specific experimental settings, could be hampered or ameliorated by a single in-network feature, however smart that single feature may be. Rather, we aim at broadly exploring a multitude of in-network features, to assess their mutual interaction and their interplay with a broad set of DAS strategies, in contrast with performance achievable with the regular TCP/IP stack.

IV. Methodology

A. Architecture

We depict in Fig. 1 the reference architecture we consider to compare TCP/IP pull-push and ICP/NDN pull-pull approaches in a DAS scenario. We focus on the open source

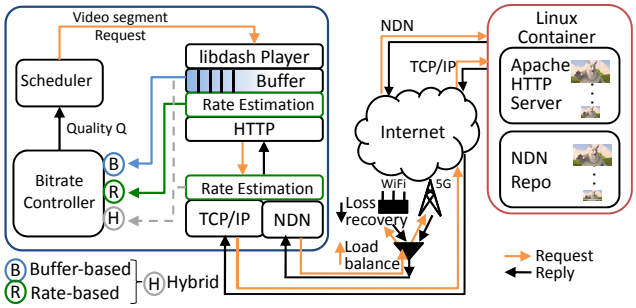


Fig. 1. Synoptic of the DASH ABR video streaming architecture used for the ICP/NDN vs TCP/IP comparison.

MPEG-Dynamic Adaptive Streaming over HTTP (DASH) [45] as the default streaming technique for our emulations, and H.264/MPEG-4 [46] for the video coding standard. We release the code we used for our testbed (as well as scripts to reproduce our experiments) as open source software at [14], and we plan to make a portion of our infrastructure available as a service for the community in the near future.[2]

Client and server. The client *controller* drives the video-segment request process, which consists in a series of video-segment requests, encapsulated in HTTP request/response pairs (orange/black arrows). As previously indicated, we select state of the art representative for all possible controller classes, namely buffer-based (PANDA) rate-based (BOLA), or hybrid (AdapTech) depending on the variable used by the controller, of which we perform a thorough calibration in Sec. V. Clients have the option to use two alternative network stacks: the TCP/IP and the ICP/NDN one. In the former case, the video is served by an Apache HTTP daemon, while in the latter case by an NDN repository.

Congestion control. As far as client is concerned, some differences are worth pointing out among these two stacks. In the TCP/IP case, congestion control of video-segment transmissions is exerted by the server, and behaves according to the well known Cubic TCP flavor. In the ICP/NDN case, control over the video-segment transmissions is exerted by the client, by means of Interest control. Since in the NDN world there is neither a TCP equivalent, nor a protocol considered as the de facto "default", we resort to use ICP [19]. Shortly, similar to TCP, ICP uses an AIMD[3] mechanism to control the window growth; however, unlike most TCP flavors, the window dynamics are regulated according to delay measurement. Also, unlike TCP, ICP does not support neither FastRetrasmit (so that it recovers losses via timeouts) nor slow-start (so that it starts with AIMD congestion avoidance). Given these different protocol characteristics, we need to assess to what extent the performance gap between TCP/IP vs ICP/NDN relates solely to the TCP vs ICP differences, which we address in Sec. V.

Bandwidth estimation. Additionally, notice that while buffer

[2]The IaaS part is not available yet, at least at submission time; but work in this direction is already in progress

[3]Cubic is MIMD, but this does not play a difference as we shall see.

level estimation is the same for both stacks, TCP/IP clients only have estimates of the download rate at video segment-level (i.e., the throughput of the TCP connection to carrying the video segment over an HTTP reply). Since bandwidth is controlled at the server side, the client cannot have finer-grained estimations out of the box (which would need support from the TCP/IP stack at server side, and an out-of-band protocol for signaling). This mismatch does not appear in the NDN case, where the local client stack can leverage NDN-chunk level information to issue finer-grained bandwidth estimates. We study bandwidth estimation granularity in Sec. VI-B.

In-network loss recovery. Finally, for what concerns in-network support, we are not the first to remark that the ICP/NDN model offers new opportunities [12] for the deployment of an efficient video streaming service, especially in mobile environments [5]: since NDN fosters both the use of caches inside nodes, and a security model where contents themselves are secured instead of the client-server connection, *Data* packets could be, in principle, retrieved from multiple locations (i.e., multipath support) and from any node in the network (implicitly building a multicast-transmission tree).

Letting large and long-lasting NDN caches aside for the sake of a fair comparison against TCP/IP, an additional advantage of NDN over IP concerns the fact that even small buffer memories can be used as temporary caches, which is very useful for *wireless loss detection and recovery* (WLDR) [22] of NDN data packets at the first hop, that are much faster than a retransmission from the server, and also consume less network resources. We investigate the impact of WLDR in Sec.VI-A.

Multi-path support. Since TCP/IP only supports a connection oriented mode, multi-path support must be enforced at application level; at the same time, we are not aware of any DASH video controller explicitly supporting multiple paths. Similarly, whereas Multi-path TCP (MTPTC) deployment is growing, a number of studies [47, 48] points to MPTCP as actually *harming* user experience. Conversely, the ICP/NDN model allows a very simple mean to support for multiple path, which can be implemented at NDN-chunk level as a simple *Load Balancing* (LB) function among all available faces, and, thus, applied directly by the client. This function remains transparent to the application, with the controller still operating on the aggregate rate. Notice that the load balancing is applied to the interest packets, but due to NDN symmetric routing where data follows back the trail of breadcrumbs left in the PIT by Interest packets, the load balancing also applies to the corresponding video Data as a result. Additionally, we consider two granularities for the LB function: namely at transport-segment (easy in NDN, but hard in TCP) vs video-segment level (possible in both NDN and TCP), that we study in Sec.VI-C

B. Scenario description

Video sources. In this paper, we use two different videos:

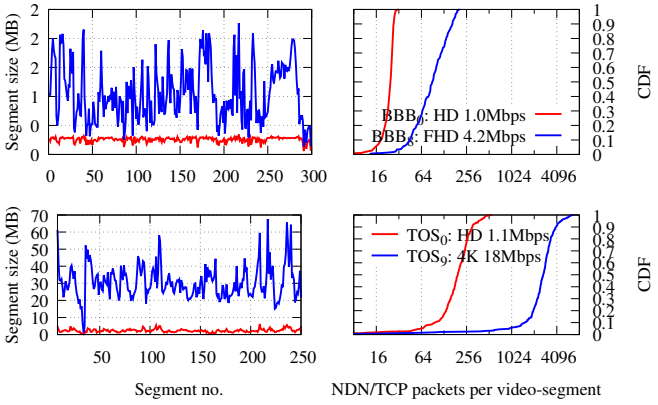


Fig. 2. Highest and lowest quality representations for the BBB (top) and TOS (bottom) video: temporal evolution of video segment size (left) and cumulative distributin function of the number of TCP/NDN messages per video-segment.

Big Buck Bunny (BBB) and Tears of Steel (TOS), which can be both found in the dataset of [49]. We are only interested in high-quality streaming and discard bitrates lower than 1Mbps, which leaves us for BBB with a total of 9 video representations, 3 of which at 1280x720p HD resolution (1, 1.2 and 1.5 Mbps) and the rest at 1920x1080p FHD resolution (2.1, 2.5, 3.1, 3.5, 3.8 and 4.2 Mbps). Similarly, for TOS video we only consider bitrates higher than 1Mbps, selecting 7 representations from the dataset, namely 1280x720p (1.1, 1.5 and 2.4Mbps) and 1920x1080p (3, 4, 6, and 10Mbps). Aiming at supporting even higher quality, for the TOS video we encoded and dashed 3 new representations, appending three new qualities 1920x1080p (FHD, 12Mbps), 2560x1440p (QHD, 15Mbps) and 3840x2160p(UHD or 4K, 18Mbps) to the existing ones, for a total of 10 representations.

For the sake of illustration, Fig. IV-B depicts the size of the segments forming both the lowest and the highest representation for BBB and TOS. The picture also shows the distribution of the number of *TCP segments (or ICN data packet)* per video-segments, which gives an idea of the granularity in bytes of the controller decision – notice that a video-segment is possibly carried by hundreds to some thousands packets for the highest qualities.

Network scenarios. We next define a number of increasingly complex scenarios, where we vary the video (BBB, TOS), the bandwidth (DASH profiles or emulated WiFi, LTE settings), the NDN network features (vanilla, WLDR, LB), and the controller logic and settings. DASH profiles are emulated using the Token Bucket Filter (TBF) of the Linux traffic control suite (tc), whereas WiFi and LTE are emulated using the ns3 channel models in MiniNet.

(A) Single client downloading BBB video with a single network channel with bell-shaped DASH bandwidth profile (i.e., profile 2a in [50] with 60s variations). This is the scenario we use to calibrate BOLA, PANDA, and AdapTech, in order to contrast their performance under the (i) TCP/IP vs (ii) vanilla ICP/NDN stacks (i.e., neither LB, nor WLDR). Results of this scenario are reported in Sec.V.

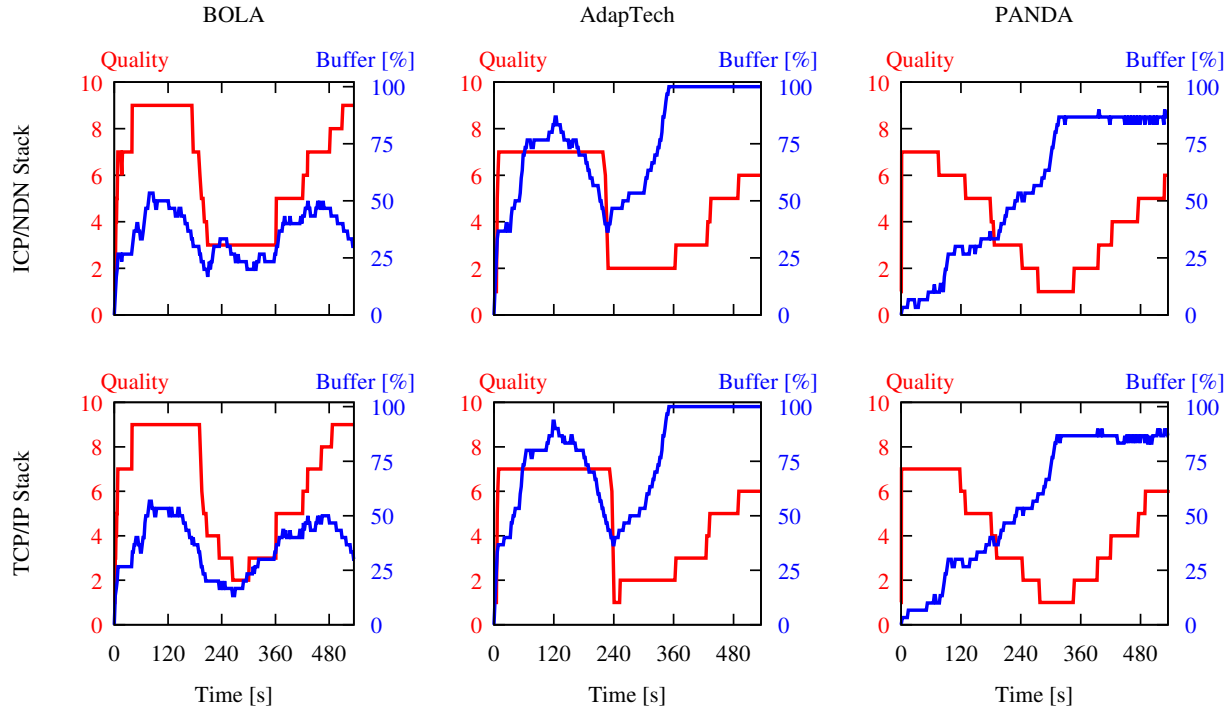


Fig. 3. Time evolution of requested quality and buffer level for the best settings of the three selected strategies (BOLA, AdapTech and PANDA, on each different columns), running on top of both an ICP/NDN stack (top) vs a TCP/IP stack (bottom). Out of the box, in simple DASH settings NDN performance matches that of TCP for all DAS strategies.

(B) Single client downloading TOS video with a single emulated WiFi channel, with the distance to the access point set to 60m, resulting in a bandwidth of approximately 6 Mbps. We contrast (i) TCP/IP against (ii) vanilla ICP/NDN or (iii) ICP/NDN with WLDR, furthermore varying the granularity of the bandwidth estimation technique at either (iv) video-segment or (v) NDN-chunk levels. Results of this scenario are reported in Sec.VI-A–VI-B.

(C) Single client downloading TOS video in a multi-homed WiFi + LTE settings. The WiFI AP is as in scenario B, whereas the LTE base station is placed at 1400m, resulting in a bandwidth of approximately 13 Mbps. In this scenario we add a LB beyond the WLDR capabilities, and contrast LB operations at (i) fine-grain, i.e., per interest vs (ii) coarse-grain, i.e., per video-segment level. Results of this scenario are reported in Secs.VI-C.

V. Calibration Results

In this section, we carry on a preliminary calibration of the selected DAS algorithms in TCP/IP and ICP/NDN stacks. Our goal is not to exhaustively present the full quantitative details of the sensitivity, but rather to show insights about the qualitative behavior of the strategies, and especially contrasting their performance under a TCP/IP and a barebone ICP/NDN stack (Sec. V-A), as well as performing a careful tuning of the best algorithmic settings for each strategy that will be fixed for the remainder of the experimental campaign (Sec. V-B). Scripts to reproduce results in this section are readily available at [14].

A. At a glance

We decouple our analysis by showing, at a glance, the behavior of the three DAS strategies in their best configuration – whereas we defer the details of finding these best configurations in the next section. We instrument the simple client-server scenario (A) with a client asking for video segments directly to the server through a wired link, whose available bandwidth and delay are varied according to a standard DASH profile (Sec. IV-B). The goal of introducing bandwidth and delay variations is twofold: on the one hand, we aim at illustrating the different operational points reached by PANDA, BOLA, and AdapTech; on the other hand, we aim at assessing the interplay between the DASH client adaptation logic at network (i.e., IP vs NDN), and transport layers (i.e., TCP vs ICP) under both stacks.

Fig. 3 reports, at a glance, the time evolution of the *requested quality* and *buffer level* for the three strategies (from left to right BOLA, AdapTech, and PANDA), and for the two stacks (from top to bottom, ICP/NDN and TCP/IP). The corresponding DASH capacity profile and the EWMA of the estimated throughput at the client is reported in Fig. 4. Two main messages arise from these results.

First, for this basic scenario (a) with no packet losses, no difference appears between the two stacks: each algorithm, being either prevalently buffer-based (e.g., BOLA and AdapTech) or rate-based (e.g., PANDA), behaves exactly the same, regardless of the network stack. This is especially reassuring since ICP and TCP are two similar but not identical congestion control protocols, that are furthermore exerted in opposite pull

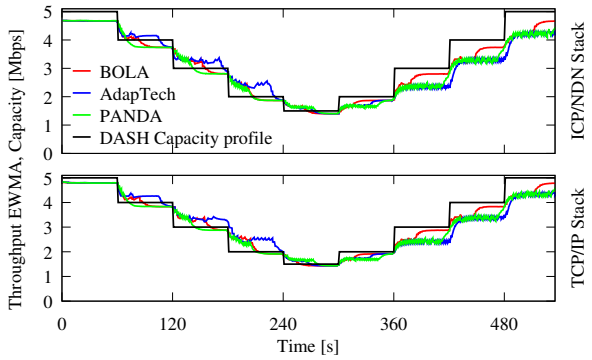


Fig. 4. Time evolution of the estimated throughput for the three selected strategies (EWMA smoothed version) and DASH capacity profile.

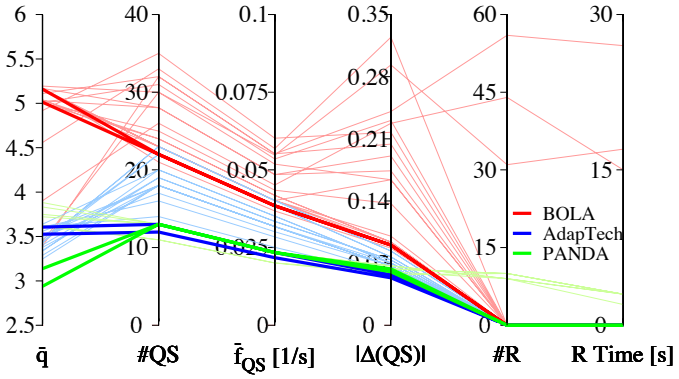


Fig. 5. Calibration of selected adaptation strategies in a simple client-server scenario with bandwidth and delay variations.

vs push modes. For instance, while both ICP and TCP use AIMD to govern the window growth, TCP reacts on losses whereas ICP reacts primarily on delay variations; additionally, TCP recovers losses mainly via FastRecovery (if the cwnd is large enough) whereas ICP recover losses via Timeouts; finally, TCP implements slow-start whereas ICP does not (in the current implementation). Still, it can be seen that transport-layer differences do not yields to noticeable changes in the DAS algorithm behavior.

Next, consider the specific behavior of each algorithm. One can clearly see a trend going from left (BOLA) to middle (AdapTech) and right (PANDA) in both the quality and buffer level. Specifically, BOLA more aggressively follows the bandwidth profile: this implies that the average quality is higher than in AdapTech and especially PANDA. As a consequence, the buffer level is lower in BOLA with respect to AdapTech and especially PANDA, since the former fully exploit the available bandwidth to download at higher quality, whereas the latter ones use the available bandwidth to increase the buffer and be more resilient against varying conditions.

B. Sensitivity analysis

Results in the previous section are gathered with DAS settings found with an empiric sensitivity analysis, which we report in this section. Specifically, we start from suggested configurations –taken from open source codebases when not available from reference papers– and vary the most prominent parameter of each algorithm.

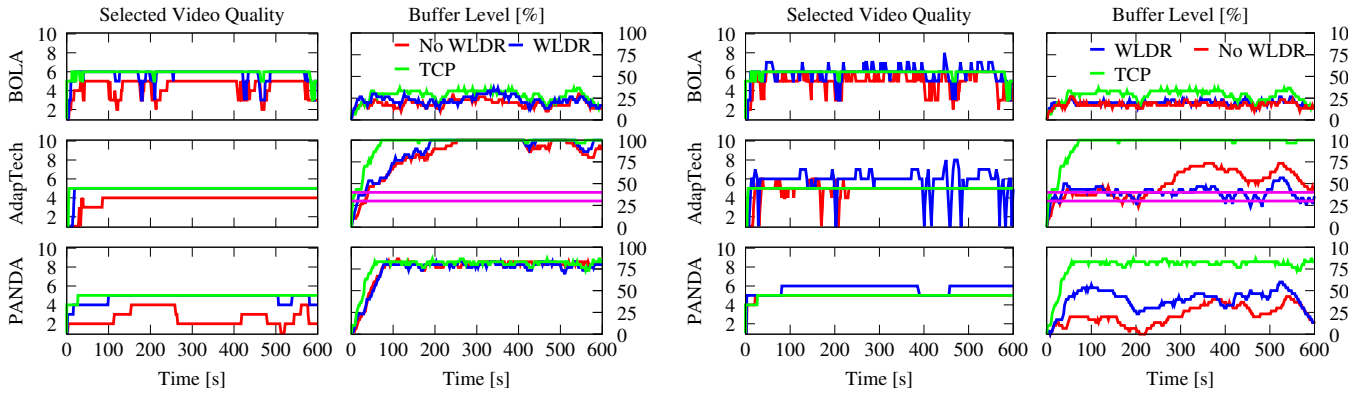
Specifically, we vary BOLA's *stable buffer threshold*, which states the difference between startup and steady state [28], in the range [6,24] seconds (the suggested default value in the DASH.js implementation [44] is 12 seconds). Concerning the AdapTech strategy , we vary the two thresholds θ_1 and θ_2 (expressed as percentage of the buffer size [29]), which affects the behavior of AdapTech in steady state, exploring $\theta_1 \in \{10\%, 20\%, 30\%\}$ and $\theta_2 \in \{40\%, 60\%, 80\%\}$ while, after preliminary investigation, we keep the amount of time needed to set the *can-switch-up* flag to the default of 10 seconds [29]. Finally, for PANDA, we tune the B_{min} parameter, which we adapt to the length of the buffer in our experiments (i.e., 60 seconds), and vary as $B_{min} \in \{34, 44, 54\}$ seconds. Additionally, we use two separate configurations: a more aggressive one, which follows the settings for the thresholds Δ_{up} and Δ_{down} suggested in in [27], while we obtain a more conservative behavior with the settings described [51].

In order to comprehensively compare the three selected DAS algorithms, we consider six different metrics:

- *Average Video Quality* $\bar{q}$: average downloaded quality over all chunks for the selected algorithm. It is computed as $\bar{q} = \frac{1}{K}\sum_{k=1}^{K} q_k$.
- *Number of Quality Switches* $\#QS$: total number of times the adaptation logic changes the requested quality.
- *Average Quality Switch Frequency* $\bar{f}_{QS}$: computed as the inverse of the average continuous quality playback (i.e., lapse of time at which successive segments are requested at the same quality), that is $\bar{f}_{QS} = 1/\frac{1}{S-1}\sum_{z=1}^{S} t(QS_z) - t(QS_{z-1})$, where $t(QS_z)$ is the time instant of the z-th quality switch, and $t(QS_0) = 0s$.
- *Average Quality Variations* $|\Delta(QS)|$: it represents the average magnitude of quality switches between consecutive segments, that is $\frac{1}{K-1}\sum_{k=1}^{K}|q_{k+1} - q_k|$.
- *Number of Rebuffering Events* $\#R$: number of times the video playout is interrupted owing to buffer depletion (i.e., rebuffering events).
- *Total Time Rebuffering* $RTime$: total amount of time spent rebuffering.

In order to succinctly represent the above 6 KPI for the combination of the 46 settings explored, we depict results as a parallel coordinate plot in Fig. 5, which allows to grasp the correlation between KPIs for specific settings. Each line in the plot corresponds to performance gathered by a DAS algorithm with specific setting: in the parallel coordinate representation, lines are a pure representation artifact that joins values taken by a specific DAS setting represented over multiples vertical axes. In particular, Fig. 5 associates a specific color to each strategy (namely red for BOLA, blue for AdapTech and green for PANDA), and solid thick lines with a brighter color designates the best combination selected for each strategy (note that two thick lines appear, that are obtained for the TCP/IP and ICP/NDN stacks for the same setting).

Results in the sensitivity confirm to a greater extent the prevalence of two complementary behaviors: a more *ag-*



(a) Coarse granularity, per video segment bandwidth estimation (TCP-like)

(b) Fine granularity, per NDN-chunk bandwidth estimation (NDN-like)

Fig. 6. Impact of in-network loss recovery and of the bandwidth estimation granularity: (a) when a coarse video-segment granularity is used (for both NDN and TCP), it can be seen that NDN+WLDR performance matches that of TCP. (b) when a per-packet granularity is used (for NDN only), it can be seen that more bandwidth can be exploited, making the protocol more aggressive and thus either better performing (PANDA) or prone to more quality switches (AdapTech).

gressive one, associated to BOLA, and a more *conservative* one, expressed by both PANDA and AdapTech. Indeed, the family of parallel curves associated to BOLA (i.e., red ones) identify, as a whole, an adaptation strategy able to provide a higher average quality ($\bar{q}$) to the detriment of rebuffering events (in some cases) and quality switches: indeed, both their number and frequency $\bar{f}_{QS}$ are, on average, higher w.r.t. PANDA and AdapTech. In addition, as it appears from Fig. 5, BOLA presents the largest magnitude of quality switches; this outcome is linked to the higher $\bar{f}_{QS}$ and to the way $|\Delta(QS)|$ is computed (i.e., since quality switches are more frequent, it is less likely that the requested quality remains the same for a considerable number of consecutive segments, which would, in that case, reduce $|\Delta(QS)|$ by adding null terms). However, in the best BOLA setting (corresponding to a stable buffer threshold of 18 seconds), drawbacks are limited: average quality is higher, rebuffering does not happen, both the number and the frequency of quality switches are significantly reduced, and their average magnitude is almost in par with AdapTech and PANDA.

At the same time, AdapTech and PANDA offer greater *stability*, i.e., (i) a better quality smoothness, measured in terms of less frequent quality shifts of furthermore smaller amplitude, and (ii) the general absence of rebuffering events – with the exception of two configurations of the aggressive versionin [27] of PANDA. At the same time, the price to pay for the increased stability of the video playout is a *smaller average quality* $\bar{q}$ with respect to BOLA. As it can be noticed from Fig. 5, varying θ_1 and θ_2 for AdapTech produces much more variability in the number of quality shifts than in the average quality $\bar{q}$, meaning that the best AdapTech configuration (i.e., $\theta_1 = 30$, $\theta_2 = 40$) is the one that minimizes $\bar{f}_{QS}$. Finally, we rule out the aggressive configuration of PANDA as it introduces rebuffering events, which we want selected strategies to totally avoid, since they represent the major factor in user disengagement [40], and we select the least aggressive version [51] with $B_{min} = 44$s as best PANDA configuration.

VI. Experimental Results

In this section, we carry on a fair comparison of DAS performance over TCP/IP, contrasted to the performance achievable on ICP/NDN by incrementally taking into considerations features as in-network loss recovery (Sec.VI-A) different granularities of the bandwidth estimation (Sec.VI-B) and in-network load balance among multiple paths (Sec.VI-C).

A. In-network loss recovery

We now emulate a realistic lossy link, using the ns3 WiFi model, considering the scenario (B) described early. In this case, whereas TCP has decades of optimizations to recover losses in an end-to-end fashion, a vanilla NDN stack poses additional challenges. Indeed, whereas the sole sender endpoint in TCP exploits duplicated acknowledgment to cope with losses, in the NDN case the data sender endpoint varies over time, and would possibly not reliably learn about losses – even piggybacking control information in subsequent Interest messages. The simplest option for an NDN stack is thus to let the application re-issue requests after a timeout. This however is suboptimal, not only as it places the burden on the DAS application, but also since a proper timeout selection is far from being trivial (notice that RTT may vary significantly since the endpoint change at anytime). A more suitable option is therefore to perform in-network loss recovery, which is especially useful for the first wireless hop. In this case, the WiFi AP (ar a STA) can detect losses and retransmit (up to a RTT) earlier than in the TCP case. Without loss of generality, we use the Wireless Loss Detection and Recovery (WLDR) mechanism described in [22].

The impact of in-network loss recovery is clearly visible in Fig.6-(a), that shows the downloaded quality (left) vs player buffer occupancy (right) for the TCP (green), vanilla NDN (red) and NDN + WLDR (blue) cases. It can clearly be seen that, for all cases the vanilla NDN underperforms with

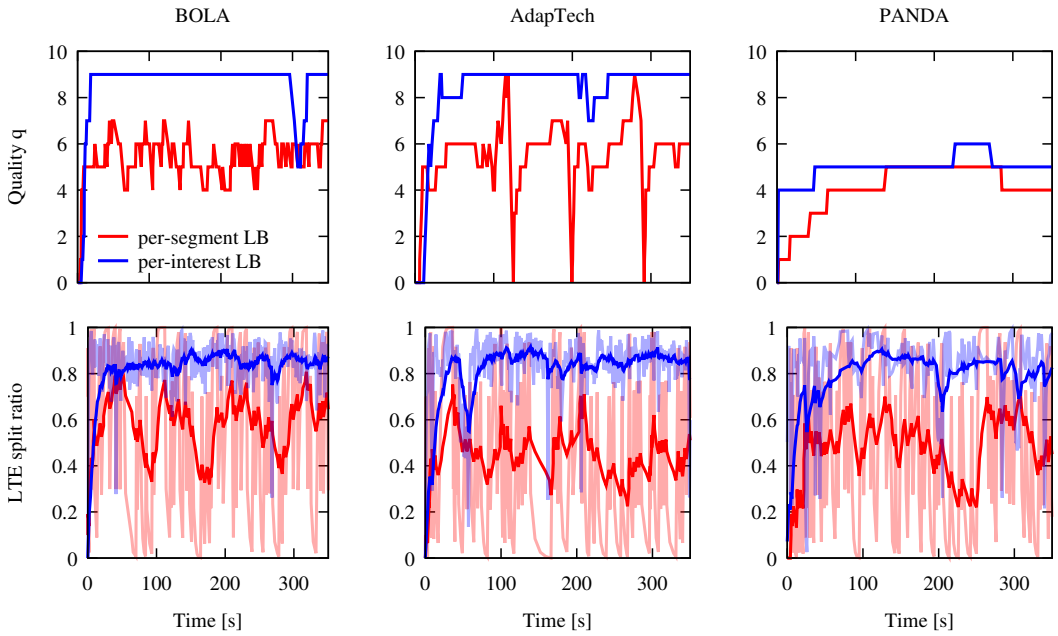


Fig. 7. In-network support: load balance among WiFi and LTE interfaces. Top plots show the instantaneous requested quality for segment vs Interest-level load balance. Bottom plots show the percentage of segments vs Interest packets aired over the LTE interface using EWMA smoothing.

respect to TCP, selecting a video quality that is consistently lower of about 1 level. Conversely, the NDN+WLDR and TCP curves are practically indistinguishable, testifying that the loss recovery mechanism, albeit needed, shall not be a prerogative of a transport-layer protocol. Additionally, consider that a TCP retransmission would traverse all the network from the originating server, unlike in the NDN WLDR case, which is thus more network-friendly.

B. Bandwidth estimation granularity

Next, compare Fig.6-(a) to Fig.6-(b) which show the impact of a coarse-grain vs a fine-grain bandwidth estimation. Specifically, each video-segment in Fig.6-(a)constitutes a bandwidth sample, whereas each NDN-chunk in Fig.6-(b) accounts for a distinct sample. It follows that the number of samples in the NDN case is much larger (by 2-3 orders of magnitude, recall Fig.IV-B), and this valuable extra information can yield a timely and refined estimation of the available bandwidth than with TCP. Notice additionally that, even in the in the most optimistic case where the TCP stack would perform such fine grained bandwidth tracking, the estimate would only be available at sender (server) side, and that out-of-band protocols should be used to signal it to the DAS client.

We consider a simple fine-grained approach that batches 30 packets to produce a bandwidth sample in NDN. While we are aware that more sophisticated approaches would be possible (e.g., packet-pair for capacity [52], train or chirps for available bandwidth [53, 54, 55], possibly in band with the data transfer [56]) our main interest here is not to quantitatively assess a specific mechanism, but to point out qualitative properties that can be expected from this building block.

Indeed, as expected comparing Fig.6-(a) to Fig.6-(b) one can notice that the instantaneous bandwidth is better tracked with fine-grained granularity. At the same time, while DAS systems can exploit more bandwidth , they also become more aggressive. The ultimate results depend on DAS algorithmic settings: in the case of PANDA, which was using more conservative settings (w.r.t. AdapTech and BOLA), the fine-grained estimate ameliorate the quality without any side-effect, unlike in AdapTech, where not only the average quality, but also the number of quality switches, increase.

Notice, in the end, that we are not advocating to indiscriminately use fine-grain bandwidth estimation: indeed, in the case of a single channel with variable performance, the ability of tracking more closely the bandwidth ultimately hurts the application by forcing an undesirable amount of quality switches. Rather, we consider more accurate bandwidth estimation as a useful building block when coupled to e.g., in-network load balance, where the availability of several (independent) channels lower the chances that all channels are poorly performing at the same time: in this case, being able to closely track channel evolution would allow to more efficiently make use of the aggregate capacity of the channels.

C. In-network load balance

We now consider the case where the client in a NDN+WLDR network is multi-homed with heterogeneous wireless technologies, specifically WiFi and LTE, both emulated via ns3. The NDN client performs load balancing of interest requests (so that data in return will travel along the trail of interests and be load balanced as well). We consider a simple algorithm [19] where clients monitor the number of Pending Interests (PI) (i.e., sent Interest packets which are not satisfied yet) for each prefix associated to a face. Any new request is scheduled with a probability that is inversely

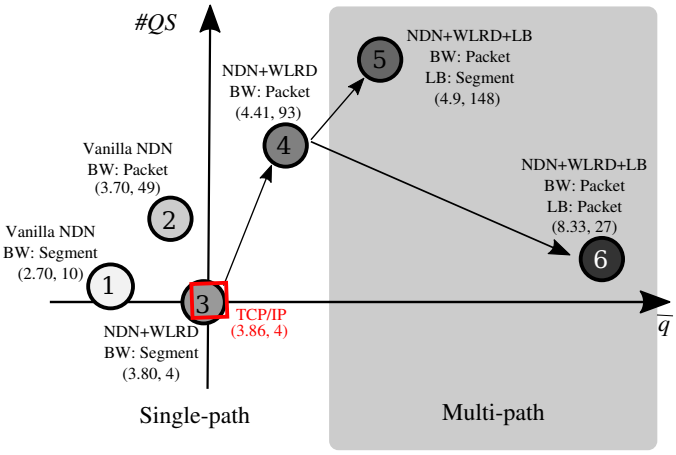


Fig. 8. Scatter plot illustrating the effect of different NDN settings w.r.t. TCP/IP for the average video quality (x-axis) and number of quality shifts (u-axis) for the AdapTech.

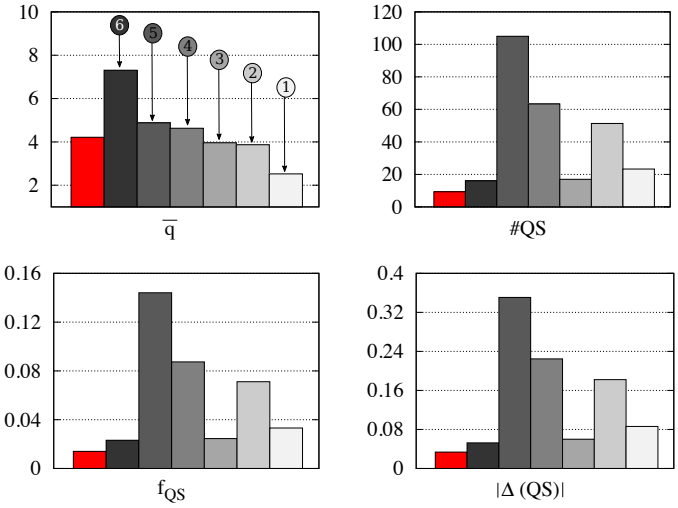


Fig. 9. Bar chart illustrating the effect of the different NDN settings (shaded gray) w.r.t. TCP/IP (red) for the average video quality $\bar{q}$, number $\#QS$, frequency f_{QS} and amplitude $|\Delta(QS)|$ of quality shifts. Bars represent averages over all PANDA, BOLA, and AdapTech DAS strategies.

proportional to the PI of that face for the matching prefix (normalized over all faces). Intuitively, a face with many PI is slow to respond, whereas a face with no PI is likely underutilized.

We do not engineer load balance on the TCP/IP case, as it would be significantly complex: this is well explained in [57], which testifies the complexity that would entail an architecture using range-requests to load balance requests at sub-video-segment level. At the same time, we argue that a TCP/IP load balance would, as for the bandwidth estimation, likely be performed at video-segment level. Since ICP/NDN+WLDR roughly matches TCP/IP performance in the single-path case (recall Sec.VI-A), we argue that ICP/NDN+WLDR with video-segment load balance would roughly match a DAS system performing segment-level load balance over multiple-paths via a TCP/IP stack.

Results are reported in Fig.7, with plots in the top row depicting the quality level for segment vs Interest level load balance, whereas plots in the bottom row report the EWMA of the split ratio of segments vs Interests sent over the LTE interface. Specifically, two curves for the split ratio are shown: the light-colored one gives more weight to the instantaneous sample ($\alpha = 0.7$) in order to gauge the variability of the split ratio, whereas the thick-colored line is an heavily smoothed version ($\alpha = 0.1$) to make the average split clearly readable.

In a nuthshell, Fig.7 shows that only Interest-level load balance allows to make use of the aggregate bandwidth, while segment-level load balancing is only partly helpful, and often even counter-productive. Notice that, by performing fine-grained load balance decisions, both BOLA and AdapTech strategies not only exhibit a tremendous gain in terms of the average quality increase, but also in terms of stability. This is due to the fact that (i) fine-grain bandwidth estimation coupled to (ii) fine-grain forwarding decisions, make these algorithms able to aggressively and promptly react to changes in the channel: additionally, the stochastic variability that negatively affected stability of the requested quality in the single channel WiFi case, is no longer a problem in this case, since channels are independent. Conversely, segment-level decisions forbid these algorithms to fully exploit the aggregate capacity, since segments are downloaded over a single channel; additionally, in case of severe channel variations, the algorithm has to finish the current segment download before switching interface, which can lead to undesirable quality switches. As previously noticed, instead, PANDA turns out to be less aggressive[4], meaning that, since it cannot fully exploit the aggregate capacity, the quality increase remains modest, even when decisions are taken at Interest-level. In this particular case, explicitly taking into account multi-path transmissions would be necessary to reach a better coupling between bandwidth estimation of the aggregate channels and requested quality.

D. Summary

Qualitative summary. We summarize the main findings of the experimental campaign with the help of Fig.8, selecting the AdapTech strategy for the sake of illustration and to avoid cluttering the picture. The picture is a scatter plot where points represent two important KPIs (i.e., the average quality $\bar{q}$ and the number of quality switches $\#QS$) for different TCP/IP or ICP/NDN configurations. In spirit of comparison, TCP/IP is set in the origin of the axes (red square), while the actual averages $(\bar{q}, \#QS)$ are also annotated in the picture. The picture shows that vanilla configurations of ICP/NDN (notably, when no in-network loss recovery capabilities are used and irrespectively of the granularity of the bandwidth estimation technique ① and ②) can hurt the performance of DAS systems: however, the use of in-network loss recovery ③ puts ICP/NDN in par with TCP/IP when the bandwidth estimation is performed at video-segment level. Additionally, an NDN sender has the opportunity of tracking more closely

[4]While it is possible to use the more aggressive PANDA settings, which can exploit the extra capacity, however this is not an angle we deem of interest, in reason of the downsides (i.e., rebuffering) early seen in the single channel scenario

the bandwidth variations, thereby being more aggressive in the requested quality, which increases both the average quality as well as the quality switch rate ④. This is expected on a single channel, whereas adding multi-path functionalities, which are very simply implemented in NDN, one can leverage statistical multiplexing to smooth out variability of bandwidth and losses. The gain in average quality is already sizable when load-balance is performed at video-segment level ⑤, which could also possibly be implemented (with some significant effort) in TCP/IP: however, the very large size of video-segments (several thousands packets at the highest quality level) may play against multi-path capabilities, still forcing undesirable quality switches. Conversely, when a fine-grained load balancing (i.e., NDN-chunk level) is used, the DAS system is able to fully exploit the available bandwidth with no penalty, i.e., almost doubling the quality with a minimal amount of quality switches ⑥ – interestingly, a packet-level technique would not be advisable in the case of connection-oriented TCP, where letting packets follow disjoint paths with different bandwidth and latency characteristics would cause significant amount of out-of-order, jeopardizing TCP congestion control.

Quantitative summary. We finally present in Fig.9, at a glance, average performance of the different NDN settings ①–⑥ just illustrated for the KPIs early used in the sensitivity analysis (with the exception of the number and duration of rebuffering events, as they do not appear with our settings). To gather results that are not tied to a specific DAS strategy, Fig.9 reports results *averaged over all DAS strategies*.

Interestingly, it can be seen that the best ICP/NDN setting ⑥ significantly increase the average quality – by almost a factor of two, even considering that quality increases in the PANDA strategy were limited (recall Fig.7). This means that one can expect consistent and considerable quantitative quality gains, that furthermore hold across strategies. Next, notice that the quality increase for ⑥ does not mechanically translate into a higher number of quality switches, which remain close to that experienced in the TCP/IP stack. As such, one can definitively confirm the interest of a carefully configured ICP/NDN stack to enhance the performance of video streaming systems in future networks: the necessary building blocks to achieve this goal are (i) fine-grained bandwidth estimation at the ICP transport layer, coupled to (ii) fine-grained load-balancing decisions among heterogeneous interfaces at the NDN client side and (iii) in-network loss recovery through the use of caches as short-term buffers.

Conversely, other NDN settings ④,⑤ lead to a more modest increases in the average quality, at the price of a significant increase of the quality switches. While a more careful evaluation of how these objective metrics translate into user Quality of Experience (QoE) is still subject to debate, we observe that a high number of quality switches may not be desirable and offset the gain in the average quality. Particularly interesting is the fact that setting ⑤ employs all ingredients of ⑥ with a single difference: i.e., the granularity of the load balancing decisions, that are taken at video-segment level. We can thus argue that the use of multiple paths could be difficult in the TCP/IP world, where decisions are likely to happen at this level of granularity [57], as this may ultimately harm user experience as remarked in [47, 48].

Finally, other naive ICP/NDN settings are less interesting as they either match ③ or even worsen ②-① performance with respect to TCP/IP. These settings correspond to a poor use of bandwidth estimation (①,③) or the lack of network support for loss recovery (①,②)

VII. Conclusion

This paper contrasts the performance achievable by adaptive bitrate video delivery using rate-based vs buffer-based adaptation logics, using either an ICP/NDN or a TCP/IP network stack. Our approach is experimental and is based on emulation of a real prototype, which we make available as open source software, along with the necessary scripts to seamlessly repeat part of our evaluation.

Our experimental campaign includes multiple videos (up to 4K resolution at 18Mbps), multiple channels (including DASH profiles, as well as WiFi and LTE access emulated via ns3) and multiple adaptation logics (PANDA, AdapTech and BOLA). As far as the ICP/NDN settings, we experiment with several building blocks that include bandwidth estimation, use of multiple heterogeneous interfaces, and in-network loss recovery. Our findings are that performance of ICP/NDN easily match and possibly significantly outperform that of TCP/IP. While this is achievable by combining relatively simple building blocks, we also find that all these blocks are *jointly* needed, and that that ICP/NDN performance can just match or even worsen with respect to TCP/IP in the other cases.

Acknowledgments

This work has been carried out at LINCS (http://www.lincs.fr). This work benefited from support of NewNet@Paris, Cisco's Chair "Networks for the Future" at Telecom ParisTech (http://newnet.telecom-paristech.fr. Any opinion, findings or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of partners of the Chair.

References

[1] C. W. Paper, "Cisco Visual Networking Index: Forecast and Methodology, 2015-2020," June 2016.

[2] ATIS 5G Americas. (2016, Dec) Understanding information centric networking and mobile edge computing. http://www.5gamericas.org/files/3414/8173/2353/Understanding_Information_Centric_Networking_and_Mobile_Edge_Computing.pdf.

[3] G. Xylomenos et al., "A survey of information-centric networking research," *IEEE Communication Surveys and Tutorials,*, vol. 16, no. 2, pp. 1024–1049, Jul. 2014.

[4] S. Lederer, C. Mueller, C. Timmerer, and H. Hellwagner, "Adaptive multimedia streaming in information-centric networks," *IEEE Network*, vol. 28, no. 6, pp. 91–96, 2014.

[5] S. Lederer et al., "Adaptive streaming over content centric networks in mobile networks using multiple links," in *Proc. of IEEE ICC*, 2013.

[6] L. Yaning et al., "Dynamic adaptive streaming over CCN: A caching and overhead analysis," in *Proc. of IEEE ICC*, Jun. 2013, pp. 3629–3633.

[7] S. Petrangeli et al., "Towards SVC-based adaptive streaming in information centric networks," in *Proc. of IEEE ICME*, Jul. 2015.

[8] R. Grandl, K. Su, and C. Westphal, "On the interaction of adaptive video streaming with content-centric networking," in *Proc. of IEEE Int. Packet Video Workshop*, Dec. 2013.

[9] D. Bhat et al., "A load balancing approach for adaptive bitrate streaming in information centric networks," in *Proc. of IEEE ICME*, Jul. 2015.

[10] D. Posch et al., "Using In-Network Adaptation to Tackle Inefficiencies Caused by DASH in Information-Centric Networks," in *Proc. of ACM VideoNext Workshop*, Dec. 2014.

[11] L. Wenjie et al., "Dynamic adaptive streaming over popularity-driven caching in information-centric networks," in *Proc. of IEEE ICC*, Jun. 2015.

[12] B. Rainer, D. Posch, and H. Hellwagner, "Investigating the Performance of Pull-based Dynamic Adaptive Streaming in NDN," *Journal on Selected Areas in Communications*, vol. 34, no. 8, pp. 2130–2140, May 2016.

[13] C. Westphal et al., "Adaptive Video Streaming over ICN," Internet Draft, https://github.com/Dash-Industry-Forum/dash.js, Oct. 2016.

[14] http://newnet.telecom-paristech.fr/index.php/icn-das/.

[15] ICNRG Datatracker Web Page. https://datatracker.ietf.org/rg/icnrg/documents/.

[16] L. Zhang et al., "Named data networking," *SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 3, Jul. 2014.

[17] V. Jacobson et al., "Networking Named Content," in *Proc. of ACM CoNEXT*, Dec. 2009.

[18] NFD, GitHub page. https://github.com/named-data/NFD.

[19] G. Carofiglio et al., "Optimal multipath congestion control and request forwarding in information-centric networks," in *Proc. of IEEE ICNP*, Oct 2013.

[20] T. Taleb and K. Hashimoto, " MS^2 : A New Real-Time Multi-Source Mobile-Streaming Architecture," *IEEE Transactions on Broadcasting*, vol. 57, no. 3, pp. 662–673, Sept 2011.

[21] G. Carofiglio, M. Gallo, and L. Muscariello, "Joint hop-by-hop and receiver-driven interest control protocol for content-centric networks," in *Proc. ACM ICN Workshop*, Aug. 2012.

[22] G. Carofiglio et al., "Leveraging ICN In-network Control for Loss Detection and Recovery in Wireless Mobile Networks," in *Proc. of ACM ICN Conference*, Sep. 2016.

[23] G. Carofiglio, L. Mekinda, and L. Muscariello, "Joint forwarding and caching with latency awareness in information-centric networking," *Computer Networks*, vol. 110, pp. 133 – 153, Dec. 2016.

[24] G. Rossini and D. Rossi, "Coupling caching and forwarding: Benefits, analysis, and implementation," in *Proc. ACM ICN Conf.*, Sep. 2014.

[25] J. Chen et al., "SAID: A Control Protocol for Scalable and Adaptive Information Dissemination in ICN," in *Proc. of ACM ICN Conf.*, Sep. 2016.

[26] J. Jiang, V. Sekar, and H. Zhang, "Improving Fairness, Efficiency, and Stability in HTTP-based Adaptive Video Streaming with FESTIVE," in *Proc. of ACM CoNEXT*, Dec. 2012, pp. 97–108.

[27] Z. Li et al., "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, Apr. 2014.

[28] K. Spiteri, R. Urgaonkar, and R. K. Sitaraman, "BOLA: Near-optimal bitrate adaptation for online videos," in *IEEE INFOCOM*, Apr. 2016.

[29] S. Akhshabi, S. Narayanaswamy, A. C. Begen, and C. Dovrolis, "An experimental evaluation of rate-adaptive video players over http," *Signal Processing: Image Communication*, vol. 27, no. 4, pp. 271–287, 2012.

[30] L. De Cicco et al., "Elastic: a client-side controller for dynamic adaptive streaming over http (dash)," in *Proc. of IEEE International Packet Video Workshop*, 2013.

[31] T. Y. Huang et al., "A buffer-based approach to rate adaptation: Evidence from a large video streaming service," in *Proc. of ACM SIGCOMM*, Aug. 2014, pp. 187–198.

[32] K. Miller et al., "Adaptation algorithm for adaptive streaming over HTTP," in *Proc. of IEEE Int. Packet Video Workshop (PV)*, 2012.

[33] C. Sieber et al., "Implementation and User-centric Comparison of a Novel Adaptation Logic for DASH with SVC," in *Proc. of IFIP/IEEE IM Symposium*, 2013.

[34] A. El Essaili et al., "Quality-of-experience driven adaptive HTTP media delivery," in *Proc. of IEEE ICC*, Jun. 2013.

[35] P. Georgopoulos et al., "Towards Network-wide QoE Fairness Using Openflow-assisted Adaptive Video Streaming," in *Proc. of ACM SIGCOMM FhMN Workshop*, Aug. 2013.

[36] T. C. Thang, H. T. Le, A. T. Pham, and Y. M. Ro, "An evaluation of bitrate adaptation methods for http live streaming," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 4, pp. 693–705, 2014.

[37] T. Y. Huang et al., "Confused, timid, and unstable: picking a video streaming rate is hard," in *Proc. of ACM IMC*, Nov. 2012.

[38] T. C. Thang et al., "Adaptive streaming of audiovisual content using MPEG DASH," *IEEE Transactions on Consumer Electronics*, vol. 58, no. 1, pp. 78–85, 2012.

[39] S. Akhshabi et al., "Server-based traffic shaping for stabilizing oscillating adaptive streaming players," in *Proc. of ACM NOSSDAV Workshop*, Feb. 2013.

[40] F. Dobrian et al., "Understanding the Impact of Video Quality on User Engagement," in *Proc. of ACM SIGCOMM*, Aug. 2011.

[41] K. K. H. Nam and H. Schulzrinne, "QoE Matters More Than QoS: Why People Stop Watching Cat Videos," in *Proc. of IEEE INFOCOM*, Apr. 2016.

[42] X. Yin et al., "A Control-Theoretic Approach for Dynamic Adaptive Video Streaming over HTTP," in *Proc. of ACM SIGCOMM*, Aug. 2015.

[43] K. Poularakis et al., "Caching and operator cooperation policies for layered video content delivery," in *Proc. of IEEE INFOCOM*, Apr. 2016.

[44] DASH Industry Forum, GitHub page. https://github.com/Dash-Industry-Forum/dash.js.

[45] I. Sodagar, "The MPEG-DASH Standard for Multimedia Streaming Over the Internet," *IEEE MultiMedia*, vol. 18, no. 4, pp. 62–67, Oct 2011.

[46] A. Puri, X. Chen, and A. Luthra, "Video coding using the H.264/MPEG-4 {AVC} compression standard," *Signal Processing: Image Communication*, vol. 19, no. 9, pp. 793 – 849, 2004.

[47] C. James et al., "Is Multipath TCP (MPTCP) Beneficial for Video Streaming over DASH?" in *Proc. of IEEE MASCOTS Symposium*, Sept 2016.

[48] X. Corbillon et al., "Cross-layer Scheduler for Video Streaming over MPTCP," in *Proc. of ACM MMSys*, Jun. 2016.

[49] S. Lederer, C. Müller, and C. Timmerer, "Dynamic adaptive streaming over http dataset," in *Proceedings of the 3rd Multimedia Systems Conference*. ACM, 2012, pp. 89–94.

[50] DASH Industry Forum: DASH-AVC/264 Test cases and Vectors. http://dashif.org/wp-content/uploads/2015/04/DASH-AVC-264-Test-Vectors-v09-CommunityReview.pdf.

[51] L. Zhi et al., "Probe and Adapt: Rate Adaptation for HTTP Video Streaming At Scale," Jul. 2013, arxiv:1305.0510v2.

[52] R. Kapoor et al., "CapProbe: A simple and accurate capacity estimation technique," *SIGCOMM Computer Communication Review*, vol. 34, no. 4, pp. 67–78, Oct. 2004.

[53] J. Navratil and R. L. Cottrell, "Abwe: A practical approach to available bandwidth estimation," 2003.

[54] E. Goldoni, G. Rossi, and A. Torelli, "Assolo, a new method for available bandwidth estimation," in *ICIMP*, 2009.

[55] V. Ribeiro et al., "pathChirp: Efficient Available Bandwidth Estimation for Network Paths," in *Proc of Passive and Active Measurement Workshop*, 2003.

[56] P. Papageorge, J. McCann, and M. Hicks, "Passive Aggressive Measurement with MGRP," *SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 4, pp. 279–290, Aug. 2009.

[57] P. Houze et al., "Applicative-Layer Multipath for Low-Latency Adaptive Live Streaming," in *Proc. of IEEE ICC*, May 2016.