# 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 contents over the Internet is experiencing an unprecedented growth. While video permeates every application, 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. In this context, Future Internet (FI) paradigms, such as Information Centric Networking (ICN), are particularly well suited to not only enhance video delivery at the client (as in the 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 TCP/IP network stack. Our campaign, based on tools which we developed and made available as open-source software, includes multiple clients (homogeneous vs heterogeneous mixture, synchronous vs asynchronous arrivals), videos (up to 4K resolution), channels (e.g., DASH profiles, emulated WiFi and LTE, real 3G/4G traces), 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, benefits of ICN-based streaming, warning about potential pitfalls that are however easy to avoid.**

## I. Introduction

There is no doubt about mobile video predominance in future traffic trends: e.g., Cisco VNI forecasts that more than 80% of IP traffic will be video, and two-third of Internet traffic will be generated from wireless mobile devices by 2020 [1]. Traffic growth goes hand in hand with evolving video services (e.g., UHD 4K-8K, 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-channels broadcast TV to adaptive streaming over an increasingly heterogeneous 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 application-network interaction, 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 architectures 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 out 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.

The rest of the paper first provides background material on ICN, and overviews the DAS literature over TCP/IP and ICN (Secs.II–III), to select those controllers we use throughout the paper. It then introduces the architecture, the emulation platform and scenarios (Sec.IV), reports experimental results (Sec.V–VI), and finally summarizes the main lessons (Sec.VII).

## II. Background on ICN

Information Centric Networking (ICN) architectures, with their distinctive features like pull-based approach, in-network caching, natural support for mobility, multi-cast, and multi-path communications, 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). While it is out of the scope of this paper to provide a comprehensive survey of ICN (for which we refer the reader to [3]), in this section we briefly review ICN characteristics at the light of DAS requirements, and introduce the potential advantages in adopting ICN for DAS video delivery that we will later experimentally investigate.

**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 *Interest* 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), storing faces *Interest* packets originate from. NDN routers also have the capability to locally store *Data* packets, 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). Also, PIT provides natural multi-cast support, as subsequent *Interest* packets for the same content aggregate locally in the PIT but are no longer forwarded.

**Connection-less pull-based transport.** NDN leverages a pull-based transport, where rate and congestion are controlled by the receiver, similarly to DAS, where clients decide rate adaptation. *Interest* packets 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 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]. ICP robustness to mobility/path disruption and receiver-controlled multi-path are particularly useful 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 PIT requests enables fully distributed in-network decisions that may help rate, loss, mobility, and congestion control management, otherwise performed at the consumer side only. First, the content-awareness provided by names naturally enables multicast via *Interest* aggregation at the PIT. Second, temporary caching of in-transit *Data* packets extends the use of buffers from loss avoidance (to absorb input/output rate unbalance) to *reuse* (subsequent requests for the same *Data* can be served locally from the cache) and *repair* (packet loss can be recovered in the network, with no need for the sender to identify and retransmit the lost packet).

In-network control mechanisms (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]) can then significantly improve an ICP/NDN DAS over the current TCP/IP DASH: this paper studies the most useful of these NDN building blocks, illustrating and quantifying the benefits they bring over TCP/IP, as well as downsides their careless use might introduce.

## III. Background on DAS

Most of the DAS literature, with few exceptions [34, 35, 44, 45], has focused on *application-level and client-side* adaptation of the requested video quality [26]–[33], [42], and, more recently, on their systematic comparison in mobile networks [46]. Other contributions started, also, to appear, which additionally consider *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)**, 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

[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*.

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.

the proposed DAS strategy (or to carry out 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. Specifically, we select Probe AND Adapt (PANDA) [27] (mostly RB) and Buffer Occupancy based Lyapunov Algorithm (BOLA) [28] (mostly BB), as they are very popular and often used as reference benchmark in the literature, and AdapTech [29], which provides an equal balance between BB and RB classes.

### 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 clients 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 up-shift, $\Delta_{up}$, and down-shift, $\Delta_{down}$, 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 other rate-based players, PANDA is shown to have the best stability-responsiveness trade-off, for which we select it as representative RB 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 is 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 $\gamma$ 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 minimize the number of quality shifts (i.e., since a bitrate capping is introduced by monitoring the available bandwidth, utility can be sacrificed), or maximize the utility (with more quality variations). BOLA is the default strategy implemented in the DASH.js player [47], which makes it a good candidate for BB.

**AdapTech.** A stronger coexistence between BB and RB decision processes is present in AdapTech [29]. The main aim is to stabilize the buffer level around a target value, $B_{max}$, while keeping the quality as smooth as possible (i.e., avoid reacting to short term bandwidth spikes, and avoid rebufferings). 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 single-stack client-based adaptation*

As previously stated, an adaptive video streaming service might take advantage, at a relatively low cost, from built-in features of ICN [48]. 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 [5]–[11]. 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 time, 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]. Possible solutions propose to increase the decisional and computational power of intermediate nodes (e.g., by altering the media description according to cached bitrates or transcoding the cached qualities [8], or by letting ICN routers perform some form of access control [10]). However, when the in-network adaptation envelope is pushed too far, scalability issues may be encountered (e.g., as in [11], where the orchestrating entity has to solve an Integer Linear Programming (ILP) optimization problem).

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. To the best of our knowledge, systematic comparison is very rare already in the TCP/IP DASH world, where [46] represents the most notable exception. As such, the broader picture of a systematic DAS comparison under both TCP/IP and ICP/NDN stacks that this work addresses is still totally unexplored.

## 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 MPEG-Dynamic Adaptive Streaming over HTTP (DASH) [49] as the default streaming technique for our emulations, and H.264/MPEG-4 [50] for the video coding standard. We release the framework we used to orchestrate the experimental campaign as part of a Linux foundation project [51] as well as scripts to reproduce our experiments at [14]. Due to lack of space, we are unable to fully detail all relevant aspects of the framework and testbed, which we make available as an external technical report [52] for the interested reader.

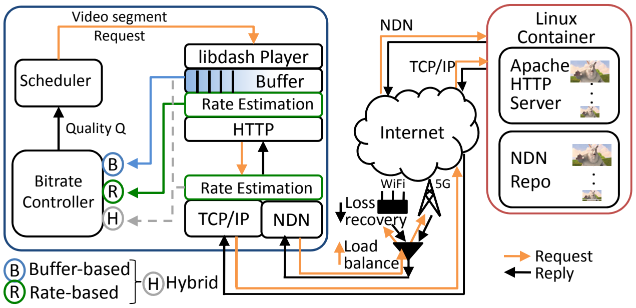


Fig. 1. Synoptic of the DAS video streaming architecture used for the ICP/NDN vs TCP/IP comparison.

**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 representatives for all possible controller classes, namely rate-based (PANDA), buffer-based (BOLA), or hybrid (AdapTech), 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 ones. In the former case, the video is served by an Apache HTTP daemon, while in the latter case by a NDN repository [53]. We consider both *single* and *multiple* clients scenarios, in both *homogeneous* and *heterogeneous* settings, with either *synchronous* or *asynchronous* start times.

**Congestion control.** Despite investigation of the TCP/ICP congestion control flavor is not among our main goals, it is worth pointing out some differences among the two stacks under investigation. In the TCP/IP case, congestion control of video-segment transmissions is exerted by the server according to the well known Cubic TCP flavor. In the ICP/NDN case, control over 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" one, we resort to our own previous work [19]. While, due to lack of space, we refer the reader to [19] for details of the protocol, it is sufficient to say that ICP uses an *AIMD* mechanism to control the window growth, which is regulated according to *delay* measurements – hence we expect ICP to be no more aggressive than MIMD and loss-based TCP Cubic. 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 differences, we need to assess to what extent the performance gap between TCP/IP vs ICP/NDN relates solely to them, which we address in Secs. V-B and VI-A.

**Bandwidth estimation.** Additionally, notice that while buffer 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 TCP/IP concerns the fact that even small buffer memories can be used as temporary caches. This would enable *wireless loss detection and recovery* (WLDR) [22] of NDN *Data* packets at the first hop – being much faster and cheaper, in terms of network resources, than server retransmissions. We investigate the impact of WLDR in Sec.VI-A.

**Multi-cast/Multi-path support.** In NDN, multi-cast and multi-path functions remain transparent to the application, whose controller still operates on the aggregate rate. Unlike in IP, NDN naturally supports multicast via PIT aggregation (and caching). Additionally, 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 DAS video controller explicitly supporting multiple paths. Similarly, whereas Multi-path TCP (MTPTC) deployment is growing, a number of studies [54, 55] 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. Notice that the load balancing is applied to *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 consequently applies also to the corresponding video *Data* packets. 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). We report multi-cast and multi-path results respectively in Secs.V-C and VI-D.

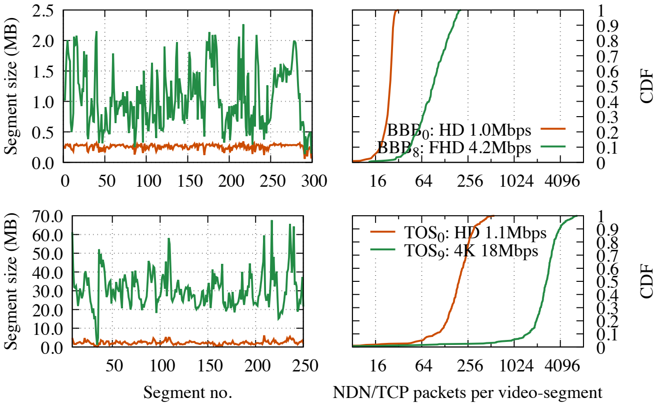


Fig. 2. Highest and lowest quality representations for the BBB (top) and TOS (bottom) video: temporal evolution of video segment size (left) and cumulative distribution function of the number of TCP/NDN messages per video-segment (right).

## B. Scenario description

**Video sources.** In this paper, we use two different videos: Big Buck Bunny (BBB) and Tears of Steel (TOS), both with a segment duration of 2s. Both videos can be found in the dataset of [56]. Being interested in only high-quality streaming leaves us with BBB encoded in 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 qualities, for the TOS video we encoded three new representations/qualities, i.e., 1920x1080p (FHD, 12Mbps), 2560x1440p (QHD, 15Mbps), and 3840x2160p (UHD or 4K, 18Mbps), that we appended to the existing ones, thus obtaining a total of 10 representations. For the sake of illustration, Fig. 2 depicts the size of the segments forming both the lowest and the highest representations for BBB and TOS. The picture also shows the distribution of the number of TCP *segments* (or ICN *Data* packets) per video-segment, which gives an idea of the granularity, in bytes, of the controller decision – notice that a video segment is possibly split in hundreds to thousands packets for the highest qualities.

**Network scenarios.** We next define a number of increasingly complex scenarios, where we vary video (BBB, TOS), bandwidth (DASH profiles, heterogeneous access), NDN network features (vanilla, WLDR, LB), and controller logic and settings. DASH profiles are emulated using the Token Bucket Filter (TBF) of the Linux traffic control suite (tc), whereas characteristics of the access network are either emulated using the ns3 channel models in MiniNet (WiFi and LTE) or enforced using real 3G/4G traces [57, 58]. Notice that, to perform a fair comparison of ICP/NDN against TCP/IP, we purposely do not consider routers equipped with caches.

Despite the ability of our framework to support the deployment of complex network scenarios, in this paper we focus on simpler topologies: the Internet cloud depicted in Fig. 1 is modeled as a simple dumbbell topology connecting the WiFi AP, or the 3G/4G base station, to the origin video server, so that we limitedly assess ICN capabilities at the network access. Notice that this simplistic setting already allows to assess implementing functions at network level, as opposite to an over-the-top approach as CDN would do in a TCP/IP case (given that WiFi AP and 3G/4G base-station are managed by the ISPs, and currently out of the reach of CDN providers). More specifically, we consider the following cases:

**(A) Calibration**: Single client downloading BBB video through a single network channel with bell-shaped DASH bandwidth profile, used to calibrate BOLA, PANDA, and

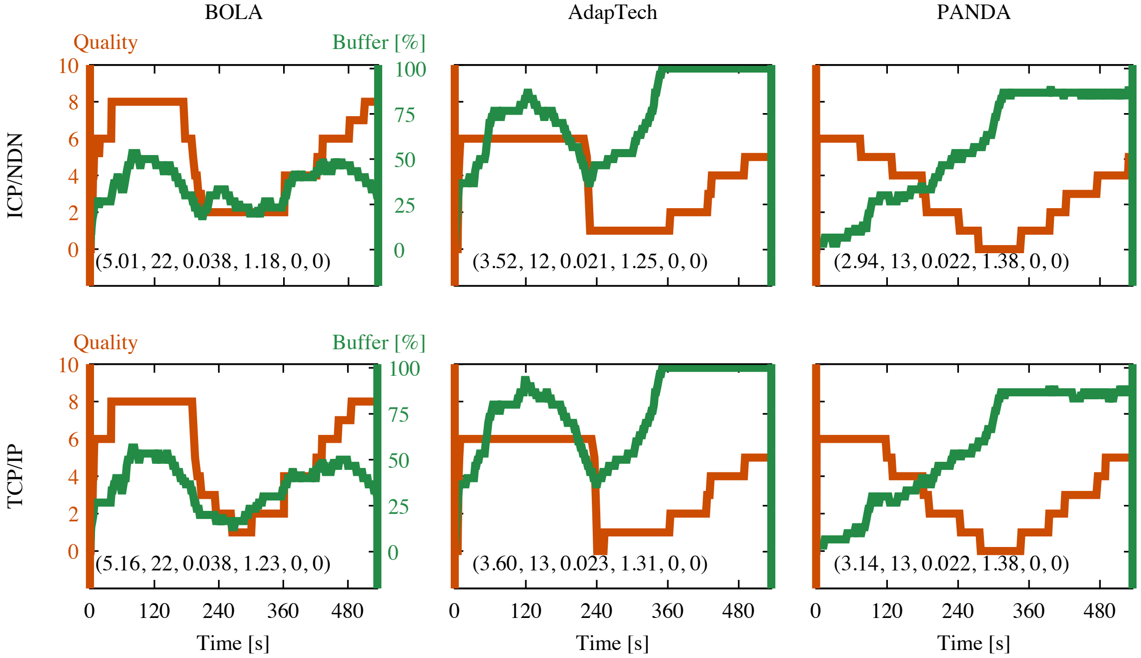


Fig. 3. Scenario (A) with BBB video: time evolution of requested quality (i.e., the correspondent video representation requested by the client, where “0” is the lowest and “8” is the highest one) and buffer level for the best settings of the three selected strategies (BOLA, AdapTech, and PANDA, on each different columns), running on top of both ICP/NDN (top) vs TCP/IP (bottom) stacks. The picture is annotated with a tuple ($\bar{q}$,$\#QS$,$f_{QS}$,$|\Delta(QS)|$,$R$,$RTime$) representing the main KPIs, namely: average quality $\bar{q}$, number $\#QS$, frequency $f_{QS}$, and amplitude $|\Delta(QS)|$ of quality switches; number $R$ and duration $RTime$ of rebufferings. Out of the box, in simple DASH settings, ICP/NDN performance matches that of TCP/IP for all DAS strategies.

AdapTech. The aim is contrasting their performance under (i) TCP/IP vs (ii) vanilla ICP/NDN stacks (i.e., neither LB, nor WLDR). Results are presented in Sec.V-B.

- **(B)** **Multi-client**: Clients downloading TOS video with *homogeneous* (either all TCP/IP or all ICP/NDN) vs *heterogeneous* (half TCP/IP, half ICP/NDN) population, where clients start time are either *synchronized* (live streaming) or *desynchronized* (VoD case). Results are presented in Sec.V-C.
- **(C)** **Transport**: Single client downloading TOS video through a single emulated WiFi channel. 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 are presented in Secs.VI-A and VI-B.
- **(D)** **Network Access**: Single client downloading TOS video, contrasting different access types and emulation techniques: model-based WiFi/LTE vs trace-driven 3G/4G, etc. Results are presented in Sec.VI-C
- **(E)** **Load balance**: Single client downloading TOS video in a multi-homed WiFi + LTE setting. In this scenario we add a LB beyond the WLDR capabilities, and contrast LB operations at (i) fine-grained, i.e., per *Interest* vs (ii) coarse-grained, i.e., per video-segment level. Results are presented in Sec.VI-D.

## V. Calibration Results

In this section, we carry out 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, 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. Scripts to reproduce results presented 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 BBB video segments directly from the server through a wired link, whose available bandwidth and delay are varied according to a standard DASH profile (namely, 2a in [59] with 60s variations). 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, and

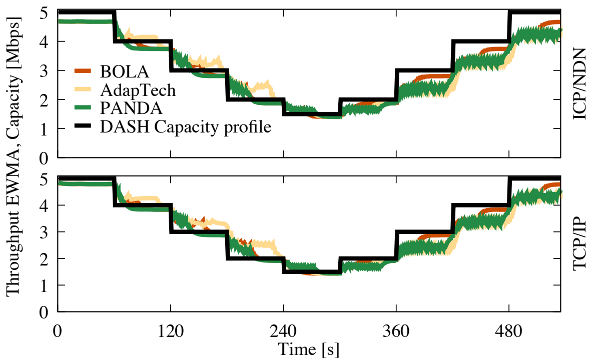


Fig. 4. Time evolution of the estimated throughput for the three selected strategies (EWMA smoothed version) and DASH capacity profile.

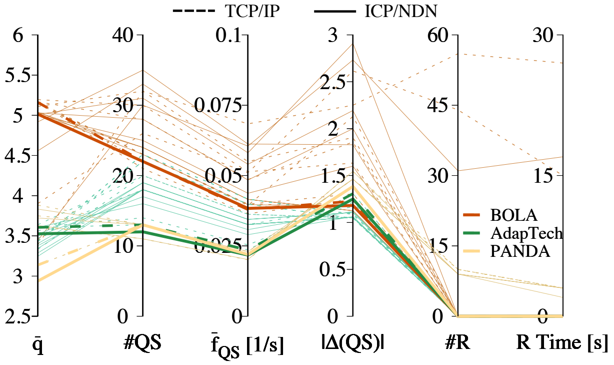


Fig. 5. Calibration of selected adaptation strategies in a simple client-server scenario with bandwidth and delay variations.

for the two stacks. 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 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 recovers 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 result in 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 results in a higher average quality than the one in AdapTech and PANDA. As a consequence, the buffer level is lower in BOLA with respect to AdapTech and PANDA, since the former fully exploits the available bandwidth to download at higher qualities, 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 parameters 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 [47] is 12 seconds). Concerning the AdapTech strategy , we vary the two thresholds, $\theta_1$ and $\theta_2$ (expressed as percentage of the buffer size [29]), which affect the behavior of AdapTech in steady state, exploring $\theta_1 \in \{10\%, 20\%, 30\%\}$ and $\theta_2 \in \{40\%, 60\%, 80\%\}$; we, instead, keep the $T$ parameter to its default value 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 $\Delta_{up}$ and $\Delta_{down}$ suggested in [27], while we obtain a more conservative behavior with the settings described in [60].

In order to comprehensively compare the three selected DAS algorithms, we consider six different metrics, among the many available, to estimate the user quality of experience [61]:

- **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 KPIs for the combination of the 46 explored settings, 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 settings: in the parallel coordinate representation, lines are a pure representation artifact that joins values taken

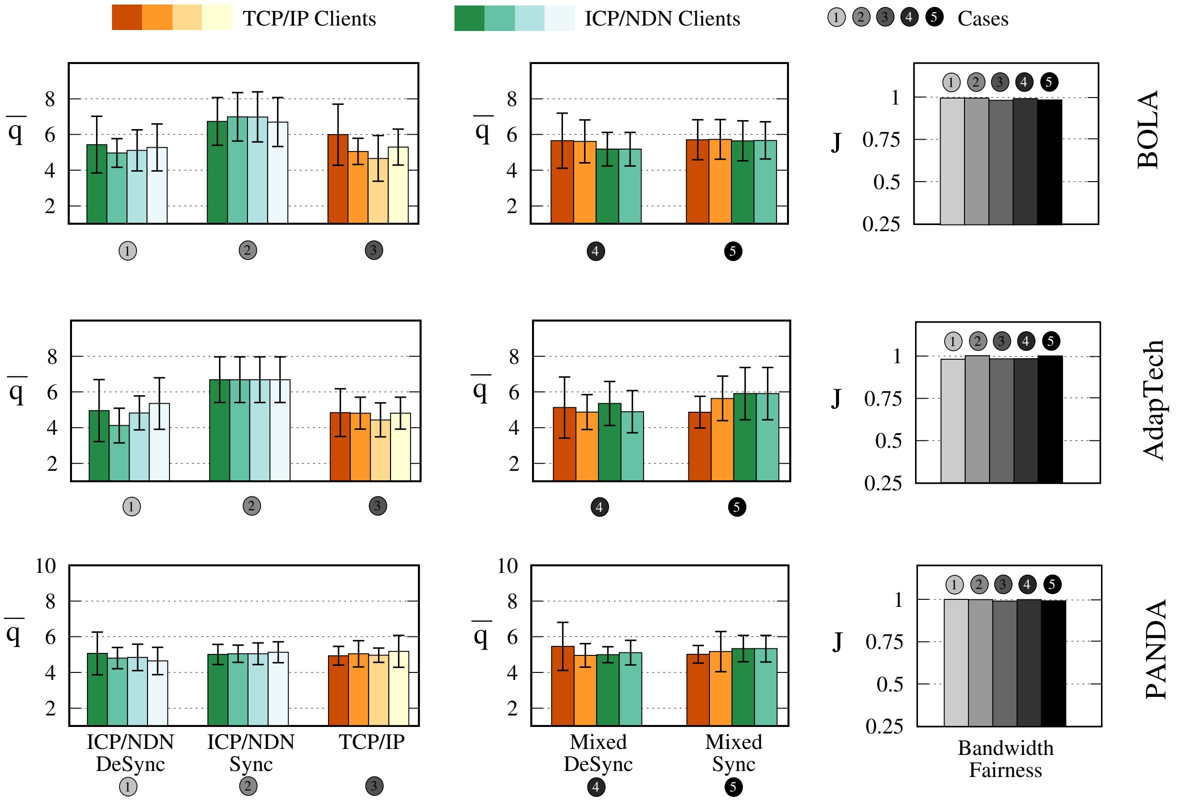
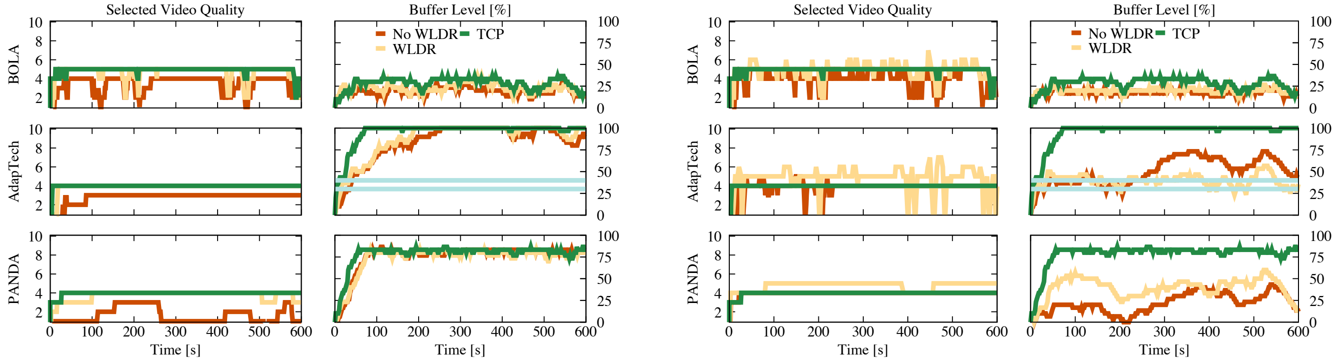


Fig. 6. Multi-client Scenarios. *Average quality* and *bandwidth fairness* (with 95% confidence intervals) under homogeneous/heterogeneous populations, with synchronized/desynchronized client arrivals for BOLA (top), AdapTech (middle), and PANDA (bottom).

by a specific DAS setting represented over multiple vertical axes. In particular, Fig. 5 associates (i) a specific color to each strategy (namely, brown for BOLA, green for AdapTech, and gold for PANDA), (ii) a specific line type for each stack (namely, dashed for TCP/IP and solid for ICP/NDN), and (iii) thicker lines with a brighter color indicating the best selected combination (note that two thick lines appear for each strategy, indicating the best combination for both TCP/IP and ICP/NDN stacks). For each pair of strategy/stack, among the settings that avoid rebufferings, we select the one that maximizes the average video quality and minimizes the number of quality switches (when the average video quality difference among two settings is within 5%, we select the setting that reduces the number of quality switches).

Results of the sensitivity reveal that there is not any relevant difference between the best cases of TCP/IP and ICP/NDN stacks for each strategy, at least in the scenario used for the calibration. They also confirm, to a greater extent, the prevalence of two complementary behaviors: a more *aggressive* 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., brown 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 18s), drawbacks are limited: average quality is higher, rebufferings do 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) better quality smoothness, measured in terms of less frequent quality shifts of furthermore smaller amplitude, and (ii) general absence of rebuffering events – with the exception of two configurations of the aggressive version of PANDA [27]. Nevertheless, 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 $\theta_1$ and $\theta_2$ 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 [60] with $B_{min} = 44$s as best PANDA configuration.



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

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

Fig. 7. Impact of in-network loss recovery and bandwidth estimation granularity: (a) when *a coarse video-segment* granularity is used (for both NDN and TCP), NDN+WLDR performance matches that of TCP. However (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).

### C. Multi-client scenarios

We next assess if the selected calibration settings yield to consistent results also in multi-client scenarios. To the best of our knowledge, the investigation of competing ICP/NDN and TCP/IP clients is a new contribution. Specifically, we include scenarios to study (i) *homogeneous* (i.e., all TCP or all NDN) as well as (ii) *heterogeneous* (i.e., half TCP and half NDN) client populations, and we further distinguish between (i) *synchronized* and (ii) *desynchronized* client arrival patterns. Simultaneous arrivals closely represent a live-streaming case, whereas asynchronous independent client requests naturally correspond to a VoD case, both of which are relevant from practical viewpoints.

In particular, we do not expect the synchronization scenario to have any noticeable effect for TCP/IP. Conversely, in the ICP/NDN synchronized case, *Interest* packets aggregate at the PIT: this is beneficial, since in case the bottleneck is upstream the access link, then we expect PIT aggregation to form a multi-cast tree, which lead ICP/NDN clients to use the bottleneck bandwidth more efficiently. The five cases we consider are reported in Fig.6. The picture reports the average quality for a homogeneous population of ① ICP/NDN asynchronous clients, ② ICP/NDN synchronous clients and ③ TCP/IP clients, as well as for a heterogeneous TCP and NDN population with either ④ asynchronous or ⑤ synchronous clients. The picture also reports (gray bars) the Jain fairness index of the bandwidth share, useful to assess if some of the $N = 4$ clients starves the other ($J \approx \frac{1}{N}$), or if clients equally compete for resources ($J \approx 1$).

Three very important takeaways can be gathered from the picture. *First*, as expected, PIT aggregation lead NDN synchronous clients to increase the quality without increasing the upstream bandwidth: this is particularly visible for the BOLA and AdapTech strategies contrasting ② against ① and ③, where at least one quality level can be consistently gained in the emulation settings. *Second*, from ④ one can easily gather that ICP/NDN appears to be no more aggressive than TCP/IP (bandwidth share is fair and quality is in par or slightly lower), which is expected due to the differences in the window growth dynamics (delay-based and AIMD in ICP vs loss-based and MIMD in TCP Cubic). *Third*, and most important, from ⑤ one can gather that previous properties combine: especially noticeable under AdapTech, the PIT aggregation makes synchronous ICP/NDN clients consume content as leafs of a multi-cast tree. This, on the one hand, improves the quality for ICP/NDN, and, on the other hand, reduces the used upstream bandwidth, which now becomes available for TCP/IP as well. Notice also that no side effects appear, as bandwidth share is still fair also under this circumstance.

Overall, we verify expected benefits of ICP/NDN to hold, and we additionally conclude our calibrated settings to be robust to multi-client scenarios as well.

## VI. Experimental Results

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

### A. In-network loss recovery

We now emulate a realistic lossy link, using the ns3 WiFi model (in order to get a bandwidth of approximately 6 Mbps, and a distance to the access point of 60m). In this case, whereas TCP has decades of optimizations in recovering losses in an end-to-end fashion, a vanilla NDN stack poses additional challenges. Indeed, while the sole sender endpoint in TCP exploits duplicated acknowledgment to cope with losses, the NDN *Data* sender endpoint might vary over time, making it difficult to learn about losses – even piggybacking control information in subsequent *Interest* messages. The simplest option for a NDN stack is thus to let the application re-issue requests after a timeout. This is, however, suboptimal, not only because it places the burden on the DAS application, but also because a proper selection of the timeout is far from being trivial (notice that RTT may vary significantly due to the possible endpoint variation). 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 (or a STA) can detect losses and retransmit (up to one RTT) earlier than in the TCP case. Without loss of generality, we use the Wireless Loss Detection and Recovery (WLDR) mechanism described in [22]. It is important to notice that this mechanism does not require additional caches, as it only leverages buffers on routers' linecards (of about 1 MB).

The impact of in-network loss recovery is clearly visible in Fig.7-(a), which reports, for all the adaptive strategies, the selected video quality (left) vs player buffer occupancy (right), both for TCP/IP (green), vanilla ICP/NDN (brown), and ICP/NDN with WLDR (gold). It can be noticed that, for all the strategies, vanilla ICP/NDN does not guarantee the same performance as TCP/IP in terms of selected video quality (which is consistently 1 level lower). Conversely, the ICP/NDN stack with WLDR equals the performance of TCP/IP, testifying that the loss recovery mechanism, albeit needed, could be easily implemented directly at the network layer in NDN, and so outside the sole transport layer (as in TCP/IP). Additionally, an in-network loss recovery mechanism (like WLDR) would reduce the communication overhead w.r.t. a TCP retransmission, which needs to traverse the whole path from client to server.

### B. *Bandwidth estimation granularity*

Figs.7-(a) and 7-(b) contrast the impact of a coarse-grained bandwidth estimation against a fine-grained one. Specifically, each video segment constitutes a bandwidth sample in Fig.7-(a), whereas a bandwidth sample in Fig.7-(b) results from averaging estimates over 50 NDN packets, as also proposed in [62]. As a consequence, the number of available samples in the NDN case can grow up to two orders of magnitude more with respect to the TCP segment-based estimate (recall Fig. 2), thus resulting in valuable extra information that can be used to implement a more timely and refined estimate of the available bandwidth. While we are aware that more sophisticated approaches than a 50 packets batch mean would be possible (e.g., packet-pair for capacity [63], train or chirps for available bandwidth [64, 65, 66], possibly in band with the data transfer [67]), 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. Furthermore, provided the availability of the complex aforementioned techniques for the TCP case, the resulting estimate would, however, be available only at the server side, requiring out-of-band protocols to signal it to DAS clients (differently from the NDN case).

As expected, comparing Fig.7-(a) to Fig.7-(b), we can notice that the instantaneous bandwidth variations are better tracked by a fine-grained estimate, allowing DAS clients to better exploit the available capacity. At the same time, a more responsive adaptation logic might result in an increased aggressiveness (as for AdapTech and BOLA, where the number of quality switches increases), if not smoothed by the logic itself (i.e., the conservative version of PANDA takes advantage of the finer-grained estimate by increasing the selected video quality, without any side effect). It is worth noticing that while we are not advocating to indiscriminately use fine-grained bandwidth estimate (i.e., see the increase in quality shifts in some cases), we consider more accurate techniques to estimate the available bandwidth as a useful building block when coupled to, e.g., in-network load balance, where the availability of multiple (independent) channels can be exploited to either increase the selected video quality, or guarantee the same quality if some of them experiences bad conditions: in these cases, being able to closely track channel evolution would allow to fully profit from the aggregate capacity.

### C. *Access technology and emulation technique*

In order to confirm findings discussed in the previous sections in more realistic conditions, we now contrast performance gathered via *emulated channel models* against those collected by using *real traces*. In particular, we use both 3G [2] and 4G [3] real traces, available at [57, 58]. We remark that if, on the one hand, emulated models have the benefits of yielding arbitrarily long stochastic processes – which ensure statistical relevance of the experiments over multiple independent repetitions, real traces, on the other hand, represent samples of finite length, but of real conditions – without requiring complex calibrations.

We report performance at a glance in Fig. 8: the bottom plot in the figure illustrates the available bandwidth for the different emulated (left) and trace-driven (right) cases. Top plots report detailed time evolution for AdapTech, and are complemented by bar charts showing average (along with 95% confidence intervals bars over 10 repetitions) for all DAS strategies. It can be seen that, despite differences in the stochastic nature of these processes, there is an agreement between the available bandwidth and the average qualities: e.g., emulated WiFi and trace-driven 3G performance are similar for all DAS strategies, and the same holds for emulated LTE vs trace-driven 4G. Overall, the comparison of both methodologies allow to conclude that performance gathered over emulated models are not only statistically relevant, but also qualitatively and quantitatively in agreement with real trace driven conditions.

### D. *In-network load balance*

We now consider the case where the client in a NDN with WLDR network is multi-homed with heterogeneous wireless technologies. Specifically, we restrict our attention to emulated WiFi and LTE conditions, which we expect to be both statistically relevant and with a sufficient degree of realism for our purposes. The distance to the WiFi access point is set as in VI-A, while the LTE base station is placed at 1400 m, offering a bandwidth of approximately 16 Mbps.

[2] http://home.ifi.uio.no/paalh/dataset/hsdpa-tcp-logs/bus.ljansbakken-oslo/report.2010-09-28_1407CEST.log

[3] http://users.ugent.be/~jvdrhoof/dataset-4g/logs/report_bus_0006.log

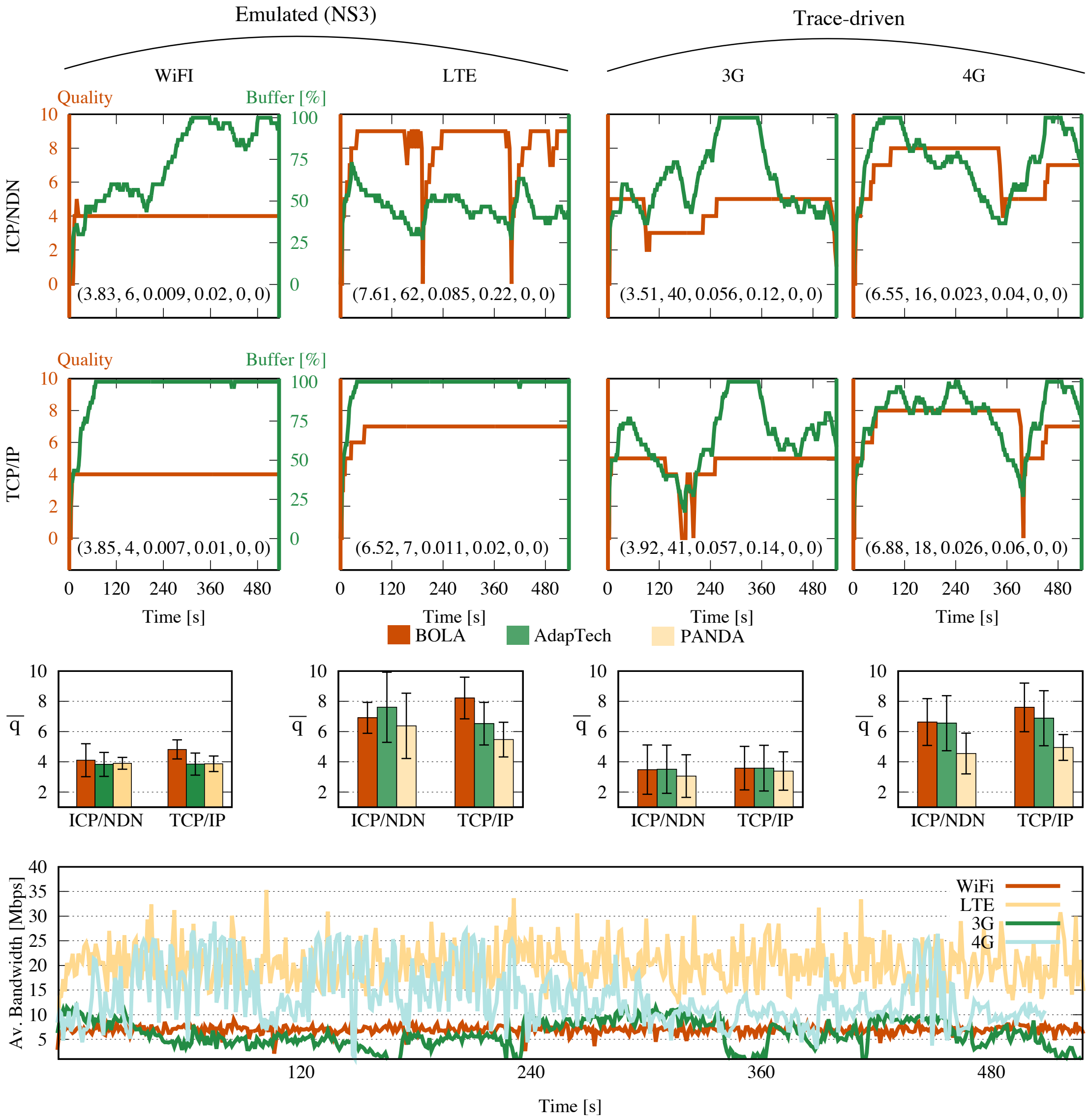


Fig. 8. Access technologies: Emulated ns3 WiFi and LTE (left columns) vs Trace-Driven 3G/4G (right columns). Top picture reports the AdapTech case as an example, annotated with $(\bar{q}, \#QS, f_{QS}, |\Delta(QS)|, R, RTime)$ performance details. Middle bar charts report average quality (with 95% CI) for all DAS strategies. Plot in the bottom row illustrates the available bandwidth with the different access technologies and emulation techniques.

The NDN client performs load balancing of *Interest* requests (so that Data packets in return will travel along the trail of *Interest* packets and be load balanced as well). We consider a simple algorithm [19], where clients monitor the number of Pending Interests (PIs) (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 proportional to the PIs of that face for the matching prefix (normalized over all faces). Intuitively, a face with many PIs is slow to respond, whereas a face with no PIs 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 [68], 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 (at a smaller implementation cost).

Results are reported in Fig. 9, 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 *Interest* packets 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 a heavily smoothed version ($\alpha = 0.1$) to make the average split clearly readable.

In a nutshell, Fig. 9 shows that only *Interest*-level load balance allows to profit from the aggregate bandwidth, while

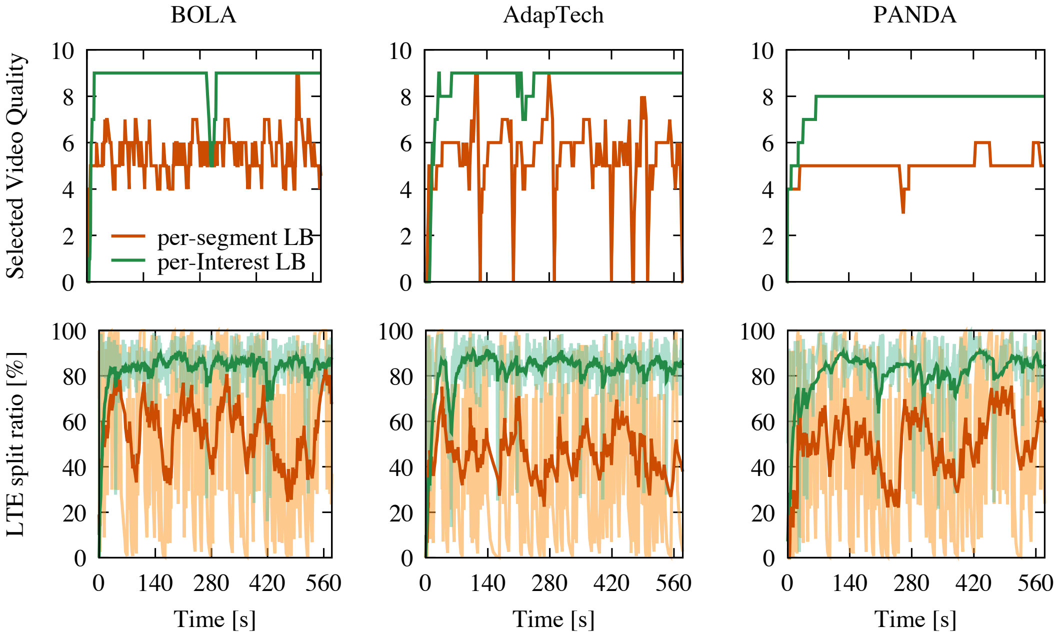


Fig. 9. In-network support: load balance among WiFi and LTE interfaces. Top plots show the instantaneous requested quality for segment vs Interest-level load balance. Bottom plots show the percentage of segments vs *Interest* packets aired over the LTE interface using EWMA smoothing.

segment-level one is only partly helpful, and often even counter-productive. Notice that, by performing fine-grained load balance decisions, both BOLA, AdapTech, and PANDA 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-grained bandwidth estimation, coupled with (ii) fine-grained 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, since channels are independent. Conversely, segment-level decisions forbid these algorithms to fully exploit the aggregate capacity, since entire segments are downloaded over a single channel; this means that, even in case of severe channel variations, the algorithm has to finish the current segment download before switching interface, thus leading to undesirable quality switches. It is worth noticing, in the end, that PANDA turns out to be the less aggressive adaptation logic[4], being in line with results shown previously. In particular, when decisions are taken at segment-level, the quality shifts are drastically reduced with respect to BOLA and AdapTech, while in the case of *Interest*-level load balancing the average quality remains constantly at one level below compared to BOLA and AdapTech.

### *E. Summary*

**Qualitative summary.** We summarize the main findings of the experimental campaign with the help of Fig. 10, 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, a NDN sender has the opportunity of tracking more closely 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

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

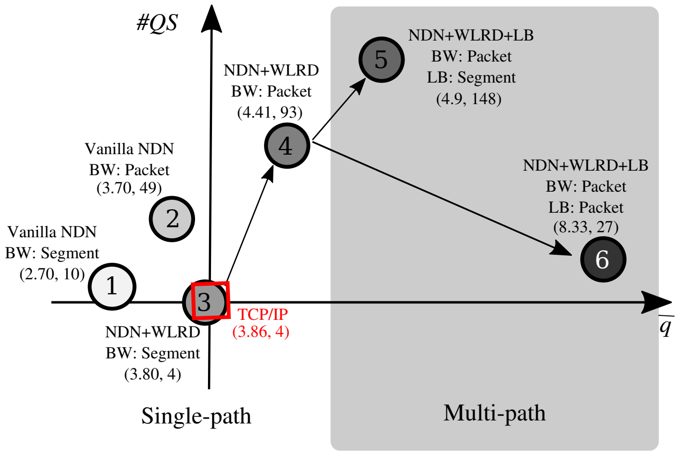
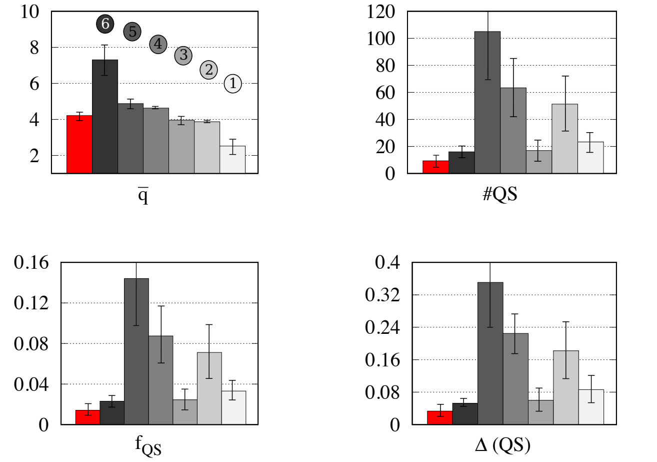


Fig. 10. Scatter plot illustrating the effect of different ICP/NDN settings w.r.t. TCP/IP for the average video quality (x-axis) and number of quality shifts (y-axis) for AdapTech.



| | TCP | ⑥ | ⑤ | ④ | ③ | ② | ① |
|---|---|---|---|---|---|---|---|
| $\overline{q}$ | 4.21 | 7.30 | 4.88 | 4.63 | 3.96 | 3.87 | 2.52 |
| $\#QS$ | 9.33 | 16.00 | 105.00 | 63.33 | 17.00 | 51.33 | 23.33 |
| $f_{QS}$ | 0.014 | 0.023 | 0.144 | 0.087 | 0.024 | 0.071 | 0.033 |
| $\Delta QS$ | 0.033 | 0.052 | 0.350 | 0.225 | 0.060 | 0.182 | 0.086 |
| $\#R$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $R\ Time[s]$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Fig. 11. Bar chart illustrating the effects of different ICP/NDN settings (shaded gray) w.r.t. TCP/IP (red) for all the considered metrics: average video quality $\bar{q}$, number $\#QS$, frequency $f_{QS}$, and amplitude $|\Delta(QS)|$ of quality shifts, number of rebufferings $R$, and rebuffering time $RTime$. We report bars representing averages over all PANDA, BOLA, and AdapTech DAS strategies, along with standard deviations and tabulated values.

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. Clearly, cases ①-⑤ are the pitfalls to be avoided in order to attain the desirable ICP/NDN operational point ⑥.

**Quantitative summary.** We finally present in Fig. 11, 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. 11 reports results *averaged over all DAS strategies*.

Interestingly, the best ICP/NDN setting ⑥ significantly increases the average quality – by almost a factor of two. This means that one can expect consistent and considerable 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 in 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 (e.g., ④ and ⑤) lead to a more modest increases in the average quality, at the price of a significant increase of the quality switches. In line with studies that model how these objective metrics translate into user Quality of Experience (QoE) [69, 70], we observe that a high number of quality switches may not be desirable since it can 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 [68], as this may ultimately harm user experience as remarked in [54, 55].

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 to 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 developed on top of an ICP/NDN or a TCP/IP network stack. Our approach is experimental and 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, or real 3G/4G traces), multiple clients (in homogeneous and heterogeneous population mixture, with synchronous and asynchronous arrival patterns) and multiple adaptation logics (PANDA, AdapTech, and BOLA). Concerning 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 ICP/NDN performance can just match or even worsen with respect to TCP/IP in the other cases.

Overall, we believe this work constitutes a first milestone towards a fair and complete assessment of fully fledged NDN video distribution systems, and their comparison with state of the art CDN technologies implemented over a classic TCP/IP stack. The following step to achieve this more ambitious goal, would be that of contrasting the two alternatives in more realistic scenarios (e.g., more complex topologies with several origin servers, multiple videos, realistic user arrival and mobility patterns, etc.). This would allow to better grasp pros and cons of the two architectures (e.g., CDN request redirection and load balancing vs NDN multicast and multipath support), as well as to assess their impact on the overall performance from the user viewpoint.

## 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.

## 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 et al., "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 et al., "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 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 et al., 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 et al., "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.

[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. Thang et al., "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] H. Egilmez et al., "An Optimization Framework for QoS-Enabled Adaptive Video Streaming Over OpenFlow Networks," *IEEE Transactions on Multimedia*, vol. 15, no. 3, pp. 710–715, Apr. 2013.

[45] K. Miller et al., "A Control-Theoretic Approach to Adaptive Video Streaming in Dense Wireless Networks," *IEEE Transactions on Multimedia*, vol. 17, no. 8, pp. 1309–1322, Aug. 2015.

[46] T. Karagkioules, D. Tsilimantos, C. Concolato, and S. Valentin, "A Comparative Case Study of HTTP Adaptive Streaming Algorithms in Mobile Networks," in *Proc. of ACM NOSSDAV Workshop*, Jun. 2017.

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

[48] M. Faran Majeed et al., “Multimedia streaming in information-centric networking: A survey and future perspectives,” *Computer Networks*, pp. –, 2017.

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

[50] 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.

[51] CICN project, wiki page. https://wiki.fd.io/view/Cicn.

[52] M. Sardara, L. Muscariello, J. Auge, M. Enguehard, A. Compagno, and G. Carofiglio, “Virtualized ICN (vICN): Towards a Unified Network Virtualization Framework for ICN Experimentation,” Tech. Rep., 2017. [Online]. Available: https://wiki.fd.io/images/4/44/VICN_technical_report.pdf

[53] NDN Repo, GitHub page. https://github.com/named-data/repo-ng.

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

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

[56] S. Lederer, C. Müller, and C. Timmerer, “Dynamic adaptive streaming over HTTP dataset,” in *Proc of ACM MMSys*, 2012.

[57] H. Riiser et al., “Video streaming using a location-based bandwidth-lookup service for bitrate planning,” *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 8, no. 3, p. 24, 2012.

[58] J. van der Hooft et al., “HTTP/2-Based Adaptive Streaming of HEVC Video Over 4G/LTE Networks,” *IEEE Communications Letters*, vol. 20, no. 11, pp. 2177–2180, 2016.

[59] 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.

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

[61] M. Seufert et al., “A survey on quality of experience of HTTP adaptive streaming,” *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 469–492, 2015.

[62] C. Wang, A. Rizk, and M. Zink, “SQUAD: A spectrum-based quality adaptation for dynamic adaptive streaming over HTTP,” in *Proc. of ACM MMSys*, 2016.

[63] 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.

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

[65] E. Goldoni, G. Rossi, and A. Torelli, “Assolo, a new method for available bandwidth estimation,” in *Proc. of ICIMP*, 2009.

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

[67] 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.

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

[69] C. Alberti et al., “Automated QoE evaluation of Dynamic Adaptive Streaming over HTTP,” in *Proc. of QoMEX Workshop*, Jul. 2013.

[70] T. Hoßfeld et al., “Quantification of YouTube QoE via Crowdsourcing,” in *Proc. of IEEE International Symposium on Multimedia*, Dec. 2011.

**Jacques Samain** received his MSc degree jointly from Ecole Polytechnique and Imperial College in 2015. He is currently doing his PhD at Cisco Systems conjointly with Telecom Paristech. His research interests focus on information-centric networking, video delivery and 5G.

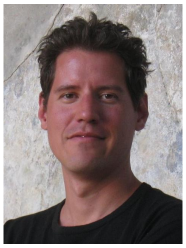
**Giovanna Carofiglio** received the MSc degree from Politecnico di Torino in 2004 and the PhD degree jointly from Politecnico di Torino and Telecom ParisTech in 2008. She currently works at Cisco Systems as Distinguished Engineer. She spent over six years at Bell Labs as head of content networking research, and was previously with the INRIA-TREC group at Ecole Normale Superieure (ENS Ulm). She was general co-chair of ACM ICN 2014, and a member of the IEEE

**Luca Muscariello** received the MSc and PhD degrees from Politecnico di Torino in 2002 and 2006 respectively. He works at Cisco Systems as Principal Engineer and is a research associate at the IRT SystemX. He spent ten years working at Orange Labs on doing research and innovation in networking. He was program co-chair of Valuetools 2013, TPC chair of ACM ICN 2014 and general co-chair of ACM ICN 2014. He is a member of the ACM and a senior member of the IEEE and SEE

**Michele Papalini** , received the MSc in Computer Science from Università di Camerino, Italy, in 2009. He received his PhD from Università della Svizzera Italiana, Switzerland, in 2015. He worked on several aspects related to information-centric networks, including architectural design, routing techniques, high-throughput matching algorithms and transport protocols. He received a best paper award in 2013. Currently he is working at Cisco Systems.

**Mauro Sardara** Mauro Sardara received his MSc degree from Politecnico di Torino in 2016. He is currently doing is PhD in conjunction with Telecom ParisTech and Cisco Systems. The main topic of his PhD is large-scale video delivery over Information-centric Networks (ICN). He is working on several research topics related to ICN, such as architectural design, transport, routing and forwarding protocols.

**Michele Tortelli** received the MSc and PhD degrees from Politecnico di Bari in 2011 and 2015 respectively. He is an Associate Editor of the Wiley Internet Technology Letters (ITL) journal, and he participated in the program committee of IEEE TVT and ACM ICN ’16,’17 conferences. He is currently a Research Assistant at Telecom ParisTech, Paris, working on design, modeling, and performance evaluation of information-oriented networks, as well as on adaptive video-streaming.



**Dario Rossi** received his MSc and PhD degrees from Politecnico di Torino in 2001 and 2005 respectively, was a visiting researcher at University of California, Berkeley during 2003-2004, and is currently Professor at Telecom ParisTech and Ecole Polytechnique. He has coauthored over 150 conference and journal papers, received 4 best paper awards, a Google Faculty Research Award (2015) and an IRTF Applied Network Research Prize (2016). He is a Senior Member of IEEE and ACM.