Supporting Mobile Applications with Information Centric Networking: the Case of P2P Live Adaptive Video Streaming

Andrea Detti, Bruno Ricci, Nicola Blefari-Melazzi

CNIT- Electronic Engineering Dept.
University of Rome "Tor Vergata", Italy
e-mail: {andrea.detti, bruno.ricci, blefari}@uniroma2.it

ABSTRACT

This paper briefly presents a P2P application for the live streaming of video contents encoded at multiple bit-rates, enabling neighboring cellular devices to increase the quality of video playback, by cooperatively using their cellular (e.g. HSDPA) and proximity (e.g. Wi-Fi Direct) wireless connections. The application is designed to exploit key functionalities of an Information Centric Network: routing-by-name, in-network caching and multicasting. We implemented a prototype of the application and assessed its performance in a test-bed based on Linux, the CCNx tool and real cellular connections.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems

Keywords

Information Centric Networking; live adaptive video streaming; cellular network; MPEG DASH.

1. Introduction

Information Centric Networking (ICN) is a concept that is recently attracting more and more research interest (see e.g. [1][2][3]). Among the proposed architectures, Content Centric Networking (CCN) [2] is probably the best-known, thanks also to its multiplatform open source implementation called CCNx [4]. In this paper we describe a P2P ICN application for live streaming of videos encoded at multiple bitrates (aka adaptive streaming). The application scenario is depicted in Fig. 1: a small set of neighboring mobile devices (peers) cooperatively download through their cellular interface segments of a live video stream, whose format is MPEG DASH [5]. Downloaded segments are shared through a proximity one-hop link (e.g., WiFi Direct) to improve the playback quality, with respect to the one they could achieve by downloading the stream independently.

ICN'13, August 12, 2013, Hong Kong, China.
ACM 978-1-4503-2179-2/13/08.

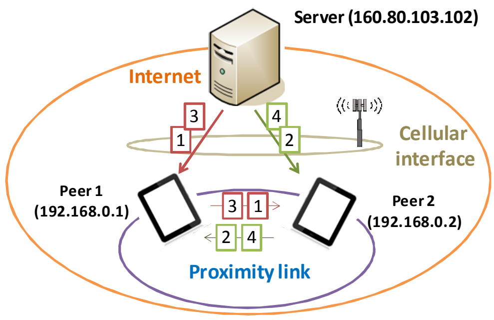

Fig. 1 – Application scenario

We assessed the performance of the application by means of an experimental test-bed based on Linux devices and real HSDPA connections. To allow the reproducibility of our results, we release the code of our application as open-source in [6]. For lack of space our description is rather concise; a longer version of the paper is available in [6].

2. Background

CCN delivers named contents, rather than providing end-to-end bit pipes. A content is any data item, e.g. a file or a segment of a video stream. CCN nodes: i) address contents by means of unique hierarchical names; ii) route-by-name content requests (i.e. Interest messages) using a longest prefix match algorithm on a name-based Forwarding Information Base (FIB); iii) route back the requested contents within Data messages and using a temporary reverse path information left in a Pending Interest Table (PIT); iv) support multicast delivery, and cache (in-network) the forwarded data.

CCN is meant for *pull* based services, where clients fetch contents from the network without caring about where data comes from. Therefore, streaming schemes suitable for natural CCN deployment are those in which video parts are pulled from clients. Thus, we decided to use the MPEG DASH video standard [5], in which the video stream is temporally structured in *segments*. A segment is an M4S file containing an interval of the video (e.g. 2s), it is uniquely identified by a name (a URLs) that includes the segment number (SN), and is available at different bandwidth (BW), i.e. coding bit-rates. In addition to M4S files, an XML Media Presentation Descriptor file (MPD) describes the video segments through meta-information such as: URLs, coding information, playback timing, etc. To play the video, a DASH player first fetches the MPD and then starts to pull and play M4S video segments from the network, following the MPD timing information and adapting the coding rate according to a local algorithm.

3. The streaming application

As shown in Fig. 1, the end-points of the streaming application are the server and the video peers. To run our experiment, we require that CCN functionality is deployed at least in these end-points; the paths linking peers to server can either be supported by plain UDP/IP sockets or by a fully fledged CCN network.

The video server contains a CCN Repository of MPD and M4S files, whose naming schemes are the following:

MPD ccnx:/server-prefix/filename.mpd

M4S ccnx:/server-prefix/filename/SN=*x*/BW=*y*.m4s

and where the server-prefix is a unique string identifying the video server. It is noteworthy that MPD and M4S files may be formed by several CCN *chunks*, whose names are automatically handled by the CCNx tool, using MPD and M4S names as prefixes. To support *live* streaming, the server also provides peers with synchronization information used during the peer joining, i.e. the sequence number of the latest published video segment.

During the video stream, peers *pre-fetch* the M4S files in batches (or *windows*) of *P* video segments at a time, and insert them in a playout buffer, emptied by a DASH player. A *pre-fetch round* starts when the source has fully published a new batch of *P* video segments and ends when the batch is fully downloaded. Peers are mildly synchronized with the source, thus they start a pre-fetch round at about the same time. During the round, a peer pulls through the cellular interface one missing segment at a time; meanwhile it tries to download all other missing segments from the CCN caches of neighbor peers using the proximity interface. Indeed, each video segment is cached by the peer's CCN layer once it has been downloaded (or even while downloading).

CCN's routing-by-name is used to enforce the download of a video segment either through cellular or proximity interfaces. Specifically, a peer has in its CCN FIB i) a static *remote-route* for ccnx:/server-prefix, whose next-hop is the remote video server (or the next CCN node of the path in case intermediate nodes are CCN enabled); and ii) the *proximity-route* ccnx:/server-prefix/filename/ SN=x/BW =y.m4s for a video segment available in the neighbor cache. These FIB entries route requests (Interest messages) of video segments not available in the neighborhood cache on the *remote-route* and requests of video segments available on peers on the *proximity-route*, due to the longest prefix matching approach.

To discover the availability of video segments in the neighborhood, a peer uses a *proximity route discovery*: as soon as a peer downloads a segment from its cellular interface, it publishes a signaling message named *Proximity-Route-Info* (PRI) that is fetched by other peers as any other content. This message contains i) the IP address and CCNx port of the peer, ii) the coding rate of the segment and iii) the estimated net cellular rate of the peer (C_i). The PRI name includes the name of the related video segment, without the coding rate component BW=*y*, and with the *control prefix* 'prd' (proximity-route-discovery). To discover the availability of the segment number *x* in the neighborhood, a peer simply tries to fetch the related PRI, whose name is ccnx:/prd/server-prefix/filename/SN=*x*.

To carry out rate adaptation, at the end of each pre-fetch round each peer computes the video coding bitrate to use in the next pre-fetch round, considering both its estimated net cellular rate and the net cellular rates of other nodes, which have been received within PRIs messages.

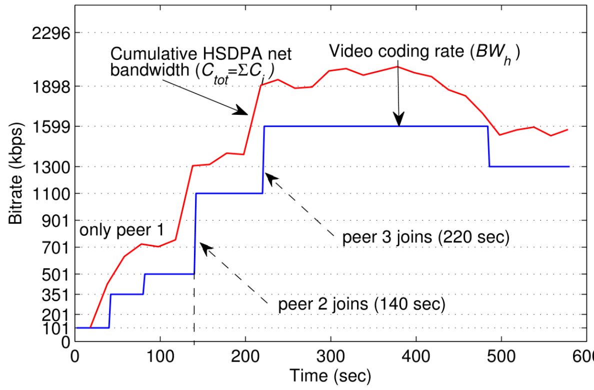

Fig. 2 – Video coding rate and cumulative net cellular (HSDPA) rate seen by peer 1 in case of three peers and P=10

Considering a video with *L* possible coding rates, and a group of *M* peers whose net cellular rates in the round *k* has been $C_i(k)$ ($1 \le i \le M$), each peer computes the coding rate BW_h ($1 \le h \le L$) of the next round ($k+1$) by solving the following maximization problem:

$$J_{i,h} = \text{floor}\left[\frac{P\ C_i(k)}{BW_h}\right]\ ;\ \max_h\left\{s.t. \sum_{i=1}^{\min(P,M)} J_{i,h} \ \geq P\right\}$$

4. Test-bed results

We implemented a Linux prototype using Java and the CCNx 0.7.0 tool. The prototype runs on three laptops, connected to the HSDPA service of the same mobile operator. The video player is VLC 2.1.0. We streamed the video "Big Buck Bunny", whose resolution is 480p; the available video qualities are fourteen, ranging from 100 kbps to 4.5 Mbps; the video is formed by about 270 segments and the segment duration T_s is 2 sec. Fig. 2 reports the coding rate of the video stream (BW_h) fetched by peers. The window size *P* is 10. The ticks of the y-axis of the plot indicate the first eleven available coding rates. The figure also shows the *cumulative* net cellular rate $C_{tot} = \Sigma C_i$. These values are measured on peer 1, which is present from the beginning of the test; instead peer 2 and peer 3 join the group at 140s and 220s, respectively. At each peer join, both C_{tot} and BW_h increase, resulting in a higher video quality experienced by each user. Obviously, the selected video coding rate is lower than the cumulative available cellular bandwidth.

5. Acknowledgements

The work for this paper was performed in the context of the FP7/NICT EU-JAPAN GreenICN project [8] and PRIN 2009 PeopleNET Project [7].

6. References

[1] T. Koponen, M. Chawla, B.G. Chun, et al. "A data-oriented (and beyond) network architecture", ACM SIGCOMM 2007

[2] V. Jacobson, D. K. Smetters, J. D. Thornton et al., "Networking named content", ACM CoNEXT 2009

[3] A. Detti, M. Pomposini, N. Blefari-Melazzi, S. Salsano, "Supporting the Web with an Information Centric Network that Routes by Name", Elsevier Computer Networks, vol. 56, Issue 17, p. 3705–3722

[4] CCNx project web site: www.ccnx.org

[5] T. Stockhammer, "Dynamic adaptive streaming over HTTP: standards and design principles", ACM MMSys 2011

[6] http://netgroup.uniroma2.it/Andrea_Detti/ICNvideo-live-DASH

[7] PeopleNET website: http://peoplenet.netgroup.uniroma2.it

[8] GreenICN website: www.greenicn.org