工學碩士　學位論文

A Testbed for Mobile Named-Data Network integrated with 4G networking devices

4G 네트워킹 디바이스로 구성된 이름 주소 기반 네트워크를 위한 테스트베드

2015 年 8 月

서울大學校 大學院

電氣・컴퓨터工學部

韓 冰

A Testbed for Mobile Named-Data Network integrated with 4G networking devices

4G 네트워킹 디바이스로 구성된 이름 주소 기반 네트워크를 위한 테스트베드

지도교수 권태경

이논문을 공학석사학위논문으로 제출함

2015 년 4 월

서울대학교 대학원

전기・컴퓨터공학부

Han Bing

Han Bing 의 석사학위논문을 인준함

2015 년 5 월

위원장 신현식 (인)

부위원장 권태경 (인)

위원 전화숙 (인)

Abstract

A Testbed for Mobile Named-Data Network integrated with 4G networking devices

Han Bing

School of Computer Science & Engineering

The Graduate School

Seoul National University

In recent years, mobile traffic (especially video traffic) explosion has become serious concern for mobile network operators. While video streaming services become crucial for mobile users, their traffic may often exceed the bandwidth capacity of cellular networks.

To address the video traffic problem, we consider a future Internet architecture: Named-Data Networking (NDN). NDN is an innovative network architecture that is being considered as a successor to the Internet. In this thesis, we design and implement framework of adaptive mobile video streaming and sharing in the NDN architecture (AMVS-NDN) with multiple wireless interfaces (e.g., 4G LTE and Wi-Fi). To demonstrate the benefit of NDN, AMVS-NDN has two key functionalities: (1) in the base situation, a mobile station (MS) tries to use either 4G LTE or Wi-Fi links opportunistically, further using Multi-Interface technology, 4G LTE and Wi-Fi links can be used simultaneously, and (2) MSs can share content directly by exploiting local Wi-Fi direct connectivity. We implement AMVS-NDN over NDN and Multi-Interface, the tests are performed in a real testbed consisting of a WiMAX base station, a LTE Femtocell and Android phones. Testing with time-varying link conditions in mobile environments reveals that AMVS-NDN achieves the higher video quality and less cellular traffic than other solutions, with using Multi-Interface, AMVS-NDN can gain the highest video quality.

Keywords : Named Data Networking, Adaptive Video Streaming, Mobile Networks, Offloading and Sharing, Multi-Interface.

Student Number : 2013-22515

Contents

List of Figures

Chapter 1

Introduction

In recent years, most of mobile network operators are facing a serious challenge due to mobile data (especially video streaming traffic) explosion [1]. While video streaming services become more crucial for mobile users, their traffic may often exceed the bandwidth capacity of cellular networks. In this situation, Named Data Networking (NDN) [2] [3], same as Content Centric Networking (CCNx), can be an attractive and efficient solution, adapting the network architecture to the current network usage pattern, i.e., data dissemination, video sharing. In NDN, every data delivery is based on the exchange of an Interest packet and a Data packet using a specific content name. Each NDN device has its own cache so that the cached data can be reused for near-future requests.

From the deployment perspective, it is a natural evolutionary path to apply NDN to wireless network environments due to highly clustered network topologies of cellular networks and wireless local area networks. For instance, it will be fairly efficient and hence necessary to satisfy requirements such like latency, of video delivery to cache popular videos at clustering points, in some situations, base station (BS) in WiMAX, LTE and access point (AP) in Wi-Fi. However, the wireless link may be fluctuating while a mobile station (MS) is moving around the coverage of multiple BSs in outdoor. Furthermore, most of currently available MSs have multiple interfaces, e.g., 3G/4G, Wi-Fi, NFC, each of which is experiencing different channel conditions. Thus, the high dynamics should be well dealt with by applications and services in the NDN architecture but currently NDN has no support for adaptive data communication [4], e.g., adaptive video streaming.

On the other hand, Dynamic Adaptive Streaming via HTTP (DASH) is a hot issue in academy and industry. The goal of DASH is to deliver video with high Quality of Experience (QoE) even in dynamic network conditions. The basic idea is that the video is encoded at multiple bit rates and resolutions, typically 7-10 different rates ranging from 150 Kbps for mobile devices up to 6 Mbps for high definition. Each encoding is divided into chunks, video segments typically between 2-30 seconds in length. The client first downloads a manifest file which contains information on the available audio and video streams, their encodings, and chunk durations. Then, the client requests one chunk of video at a time using HTTP. Depending on its rate adaptation algorithm, it detects the currently available bandwidth for the session and the video quality is adjusted accordingly. In order to support adaptive video streaming services in NDN architecture, we discuss following issues:

- **Adaptability**: The video streaming service should be aware of available connectivity and bandwidth by taking into consideration dynamic wireless link conditions and be able to adapt to the best quality of video depending the estimated bandwidth [4].

- **Video Enhancement**: Due to mobility, it always does not allow the best quality of video by only cellular networks. Thus, by exploiting different interfaces, e.g., WiMax and Wi-Fi, a MS receives video streams via a cellular link and also opportunistically accesses local Wi-Fi with other MSs to get video segments with a higher quality.
- **NDN Caching and Sharing**: In-network caching capability gives an opportunity that a MS retrieves a video segment from any nearby MS already caching that video segment, not via a BS. Furthermore, each MS can freely move and share video contents with each other [5].

Mobile service providers are deploying heterogeneous networks that augment conventional macro-cells with micro-, pico-, and femto-cells, and are utilizing a mix of radio access technologies (RATs) like 3G/4G cellular and 802.11n/ac. However, advances in wireless networking techniques have not been matched by new development of system architectures that effectively facilitate their efficient deployment. New wireless system architectures should be scalable to provide increased capacity, programmable to support new functions and advanced RATs, and self-organized. Therefore, new paradigm of mobile networking architecture should be considered.

Recently researchers are also trying to evolve current mobile backhaul networks towards the NDN infrastructure, in which, every content is with a specific name, and each network device is enabled with in-network caching, so that popular content can be cached in the intermediate network devices while being delivered, and thus successive requests can be efficiently satisfied by the cache.

In order to induce the NDN into mobile networks, we investigate current mobile networking architecture as illustrated in Figure 1.1, and find that it is very promising to implement the caching at the edges of mobile networks, e.g., eNodeB and the EPC of LTE, Wi-Fi APs, Femtocells, and so on, because they are very critical access points to aggregate user traffic and thus to efficiently utilize the caching.

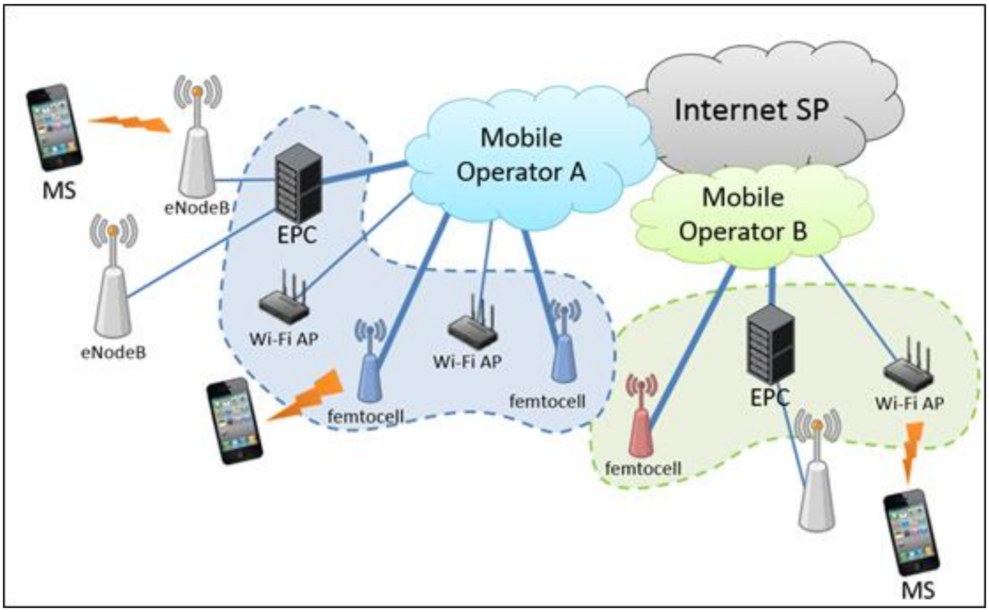


Figure 1.1 Mobile Network Architecture

Besides caching, the essential part of our work is the exploitation of the Multi-interface support for mobile devices. Poor connectivity is common when using wireless networks on the go. Connectivity comes and goes, throughput varies, latencies can be extremely unpredictable, and failures are frequent. Industry reports that demand is growing faster than wireless capacity, and the wireless crunch will continue for some time to come. Yet users expect to run increasingly rich and demanding applications on their smart-phones, such as video streaming, anywhere anytime access to their personal files, and online gaming; all of which depend on connectivity to the cloud over unpredictable wireless networks. Given the mismatch between user expectations and wireless network characteristics, users will continue to be frustrated with application performance on their mobile computing devices.

In our work, by adding functionalities for adaptive video streaming and sharing among MSs with local Wi-Fi, we realized adaptive mobile video streaming and sharing in the NDN architecture, termed AMVS-NDN. We implemented AMVS-NDN within NDN, and conducted experiments in a real test-bed consisting of a WiMAX BS and two MSs equipped with Wi-Fi. The experimental result reveals that AMVS-NDN outperforms other designs in terms of the average video quality, i.e., PSNR, and the amount of reduced cellular traffic.

Our vision requires much more than just multiple radios and multiple networks, it requires that the mobile client (as well as the applications and user) can take advantage of them. Today's clients are ill-equipped to do so, having grown up in an era of TCP connections bound to a single physical network connection. This leads to several well-known shortcomings: (1) An ongoing connection oriented flow like TCP cannot easily be handed over to a new interface, without re-establishing state; (2) If multiple network interfaces are available, an application cannot take advantage of them to get higher throughput; at best it can use the fastest connection available; (3) A user cannot easily and dynamically choose interfaces at fine granularity so as to minimize loss, delay, power consumption, or usage charges.

Through these three limitations, we implemented our prototypes (Linux and Android using NDN) to measure the performance of experiments where several network interfaces are used. Our prototype design, is purely host-based. The sending host decides which interfaces to use, and then divides outgoing traffic over multiple interfaces.

The rest of the thesis is organized as follows. We first introduce related work in Chapter 2, and explain details of the AMVS-NDN framework in Chapter 3. Then, the video sharing will be further discussed in Chapter 4. After that, the details of multi-interface will be showed in Chapter 5. We evaluate the prototype implementation in Chapter 6, and finally conclude this thesis.

Chapter 2

Related Work

2.1 Named Data Networking

NDN (aka Content Centric Networking [3]) can be characterized by two major features: routing-by-name and in-network caching. Routing-by-name enables a content, not the host, to become a first-class citizen of the network, so a single content can be retrieved from multiple locations inherently. In network caching also gives several attractive advantages such as low dissemination latency and network load reduction. At the protocol level, a user requests a content by broadcasting its interest to the network and then any content router hearing the interest and having data that satisfies it can respond with a Data packet. Data is retrieved only in the response to an interest so a single Interest packet corresponds to a single Data packet. Audio Conferencing Tool (ACT) [6] is one of pilot applications to explore the naming and real-time support of NDN for audio conferences. Instead of relying on a centralized server keeping track of information on conferences, ACT takes a named data approach to discover conferences and speakers, and to fetch voice data from individual speakers. Yet, ACT is considering only audio conferences, that is, there is no support for video conferencing or streaming.

2.2 Adaptive Video Streaming

In adaptive video streaming, e.g., Microsoft' s Smooth Streaming [7], with each chunk download, the client measures the network bandwidth and runs a Rate Determination Algorithm (RDA) to determine which bit rate to request next. Each request represents an opportunity for the client to change bit rates. When selecting a bit rate, the RDA must consider the available bandwidth, CPU processing power, screen size, and the fullness of its buffer. The RDA must balance the desire to request high-quality video with the need to prevent its buffer from draining in order to deliver the highest sustainable quality without stops or stutters. Some of commercially available RDAs were evaluated in [8] and a rate adaptation algorithm for conversational 3G video streaming was proposed by [9]. In addition, a couple of cross-layer adaptation techniques were discussed [10] [11] [12] which can acquire more accurate information of the session quality so that the rate adaptation can be more accurately made.

2.3 MS-to-MS Content Sharing

Mobile data offloading is one of candidate solutions to address mobile data explosion. According to this trend, how to reduce the amount of cellular traffic, i.e., through opportunistic data sharing between MSs, has become an important research topic. Recently, [13] exploited device-to-device communications as an underlay to LTE cellular networks for efficient content delivery. However, it was limited to a single wireless network technology while we consider multi-technology, e.g., WiMAX, LTE Femtocell and Wi-Fi, in the NDN architecture [5].

2.4 Multi Interface

In [14] [15], the varied methodologies are showed that using the multiple link interface is available for improving the quality of QoS and reduce the misalignment and influence in android phone during the transmission process. However, these experiments are not realized based on NDN environment. The [16] provided a methodology using multiple links for improving Dynamic Adaptive Streaming over HTTP (DASH) via CCNx in Mobile networks. The result shows that the media bitrate can be improved over 15% higher than the standardization of DASH. However, this method is only realized on 3G and Wi-Fi, and it isn't consider the sharing situation.

Chapter 3

AMVS-NDN Framework

3.1 AMVS-NDN illustration

Let us illustrate a scenario to explain how AMVS-NDN supports video streaming efficiently while reducing 3G/4G link traffic. Suppose two MSs A and B with AMVS-NDN functionalities will request the same video stream, and they are in the coverage of a 3G Femtocell BS. There is a video server that employs the DASH framework, which delivers the video data via the BS. MSs A and B also employ the DASH framework, so that the bit rate of the video stream can be adjusted depending on their wireless link conditions.

MS A walks around the BS while maintaining the 3G connectivity. Depending on the distance (and hence the link condition) between MS A and the BS, MS A will request (and receive) the video data with the different bit rate over time via the BS. MS B initially stays somewhat distant from the BS, and hence MS B receives the video stream with low bit rate via the BS at the beginning. After some time, MS B is in proximity of MS A, and the local Wi-Fi link between the two MSs is available (with high bandwidth). Then MS B switches to the Wi-Fi link to download the video data from MS A directly. MS B can connect to MS A by the Wi-Fi direct or tethering. Later on, MS B is close to the BS, while MS A has a poor link with the BS, and yet they still have Wi-Fi direct connectivity.

Then MS B will receive the video data via the BS, and MS A will receive the data from MS B. The illustration is shown in Figure 3.1.

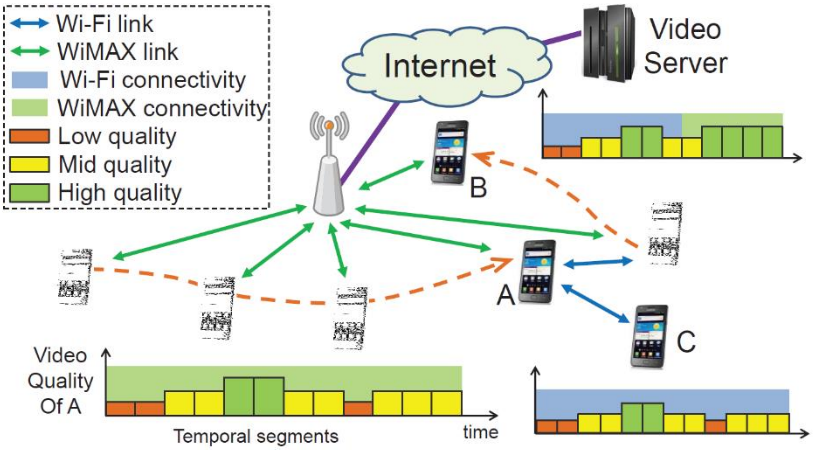


Figure 3.1 An illustration of streaming and sharing in AMVS-NDN

3.2 Video Segmentation and Naming

In AMVS-NDN, we assume that there is a metadata file for each video stream, which summarizes the stream structure of the video in terms of segments and qualities. This file is similar to the media presentation description (MPD) in the DASH framework. For a given video file name, let us name the metadata file, say "Video File Name/ INIT." The metadata file includes compression schemes, video bit rates, number of segments, size of segments for each bit rate, and so on.

For a single video source, the publisher (or its server) will maintain different copies, each corresponding to different bit rates. Also the video stream will be segmented by a specified interval, e.g., 5, 10, or 30 seconds.

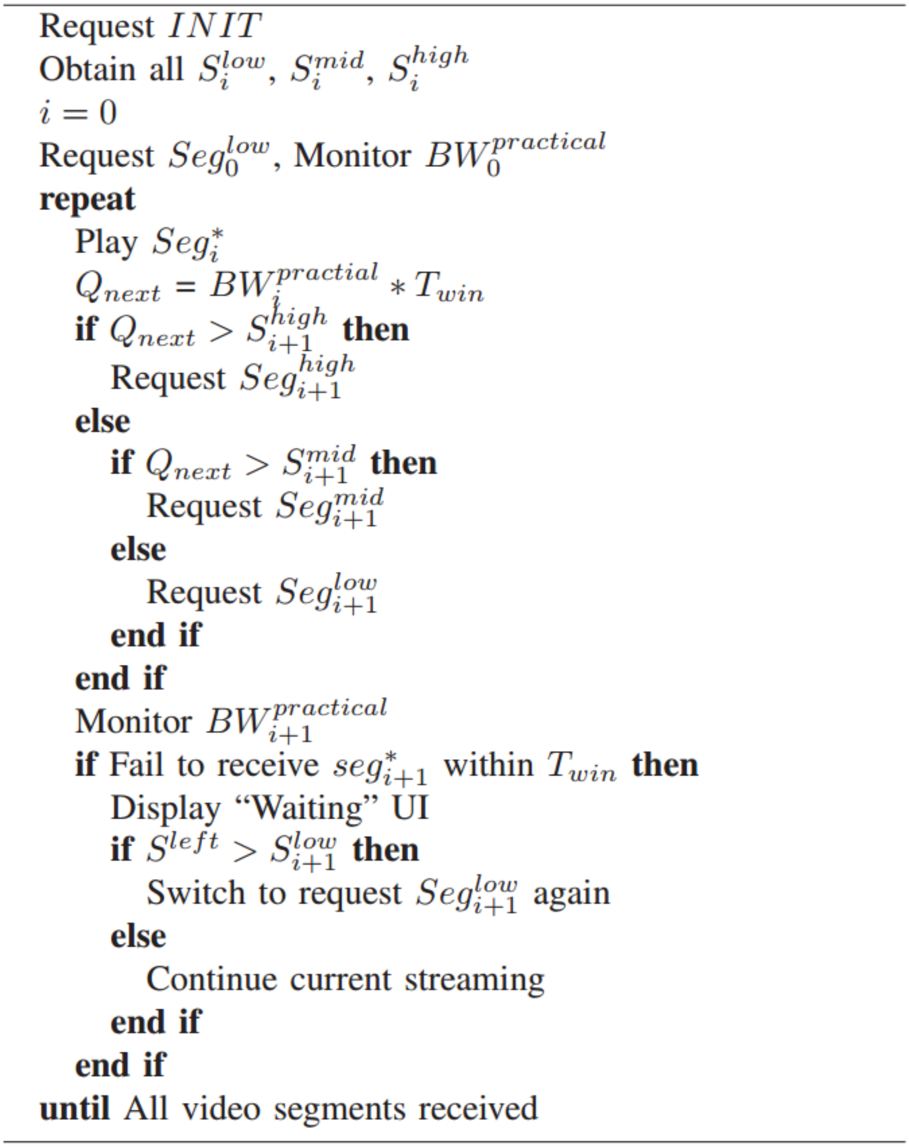
Suppose there are three video qualities for the streaming service of the final game in World cup 2014. Then the name of the 23rd video segment with low video quality can be like, **/fifa.com/video/worldcup2014/final/low/023**. The 3 name structure can be directly found out from the search engine or figured out from the metadata file.

3.3 Adaptive Streaming Strategy in AMVS-NDN

In AMVS-NDN, an MS will dynamically decide which bit rate (segment) is suitable for the current link condition and send the corresponding interest. The estimation of the current link bandwidth is based on the communication history during the past interval. The MS first obtains the INIT file, so that it can figure out the exact names of the video segments to be requested depending on the bit rate. The interest with the segment name will be routed to the video publisher, who will find the matching segment to the incoming interest.

After obtaining the INIT file, the MS will always request the first segment with low quality for conservative network bandwidth estimation, and later the segments with higher bit rates can be requested considering the effective throughput during the previous interval. Once the interests arrive the server, the corresponding segments will be sent back to the client, then the client will decode the video file and display in the screen.

The streaming application in the MS periodically evaluates the link throughput and thus decides the video bit rate for the next period. We assume that the bottleneck is always the wireless link. The length of the period highly depends on the video segmentation interval. That is, if the segmentation interval is 5 seconds, the link estimation (and the bit rate decision) period should be 5 seconds. That is, the MS always plays the video segment downloaded for the previous 5 seconds, while currently receiving the segment for the next 5 seconds. In general, at the beginning of time window i, the i^{th} video segment should be already downloaded in time window i − 1, and the $i + 1^{th}$ segment will be requested and downloaded during window i. The publisher will find matching contents stored locally to the interest; the content will be delivered back via the reversed path formed by the PITs at the nodes that the interest has passed. Assume that the segmentation/adjustment interval is Twin seconds, the procedure of throughput estimation and bit rate adjustment is shown in Fig. 1. Note that in Seg_i means the i^{th} segment, and, in Seg_i^* , ∗ indicates either low, mid or high quality version of the video stream of i^{th} segment. Likewise, S_i^* means the size of the corresponding quality version (*) of the i^{th} segment. S^{left} means the remaining bytes of the currently downloaded segment that have not been received yet. $BW_i^{practial}$ is the estimated bandwidth of the link during the i^{th} interval, and Q_{next} is the estimated link quality for the next interval.



```
Request $INIT$
Obtain all $S_i^{low}$, $S_i^{mid}$, $S_i^{high}$
$i = 0$
Request $Seg_0^{low}$, Monitor $BW_0^{practical}$
repeat
   Play $Seg_i^*$
   $Q_{next}$ = $BW_i^{practial} * T_{win}$
   if $Q_{next} > S_{i+1}^{high}$ then
      Request $Seg_{i+1}^{high}$
   else
      if $Q_{next} > S_{i+1}^{mid}$ then
         Request $Seg_{i+1}^{mid}$
      else
         Request $Seg_{i+1}^{low}$
      end if
   end if
   Monitor $BW_{i+1}^{practical}$
   if Fail to receive $seg_{i+1}^*$ within $T_{win}$ then
      Display "Waiting" UI
      if $S^{left} > S_{i+1}^{low}$ then
         Switch to request $Seg_{i+1}^{low}$ again
      else
         Continue current streaming
      end if
   end if
until All video segments received
```

Figure 3.2 Algorithm of Bandwidth Estimation and Quality Adjustment

3.4 Dealing with Delays

In the current NDN software, there is no explicit mechanism to adapt to time-varying link throughput and delay jitter. We need to address the delay variations in video streaming. In Figure 3.2, the practical link throughput changing over time is shown in the middle, which corresponds to the wireless/mobile link dynamics at the bottom. Accordingly, the video quality at the top is changed as AMVS-NDN adjusts the bit rate over time.

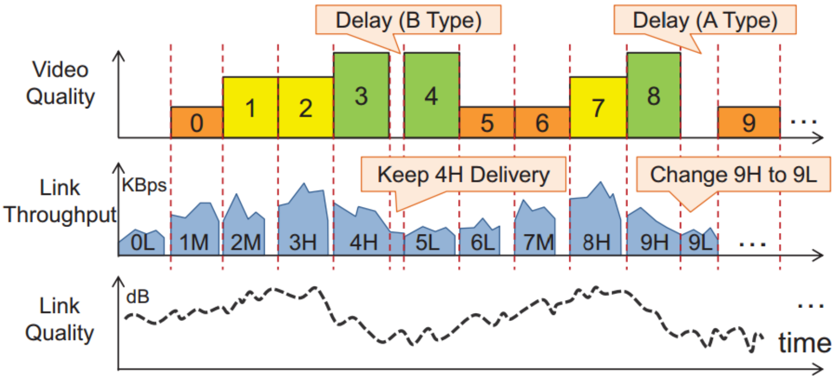


Figure 3.3 The link quality, link throughput, and playback delay in AMVS-NDN are illustrated over time

While the transmissions are going on, sometimes the link quality suddenly changes. If it gets better, transmissions of the current segment can be finished earlier than expected, which may give chances to increase the video bit rate. However, if the link quality suddenly drops, the streaming application may not be able to obtain the whole segment timely, which means we should deal with the delays.

We observe that there are two types of delays when the transmission of a segment cannot be finished within Twin: **A type** delay happens if the remaining bytes of the current segment is larger than the segment size of the low video quality, the next video quality is degraded to low to reduce the 4 disruption in the playback. We interrupt the current transmissions with I/O error exception of NDN input stream in NDN API and transmit the interest for the low quality segment immediately. At the top of Figure 3.2, segment 9 is delayed, which is A type delay. Thus the current transmission is terminated and the video quality is changed to low. **B type** delay happens if the remaining bytes of the current segment is smaller than the size of low video quality, the application shows the waiting icon in the screen and the current transmission continues. In Figure 3.3, segment 4 is delayed as B type, and hence we keep the transmission going.

The delay is mainly because of a sudden change of link quality while downloading the video segment for the next time window. Thus, initial buffering of 2 or 3 segments can mitigate the problem, but may incur long startup delay and require any complicated RDA. The tradeoff between the initial buffering and the jitter resilience may depend on the level of link dynamics and user mobility, which we will work on as future work.

Chapter 4

Video Sharing in AMVS-NDN

In the NDN architecture, each node has a cache for storing and sharing. Thus in wireless NDN, once an MS obtained video segments from the video server via the BS, other MSs nearby can opportunistically receive the video segment from the MS who holds the segments.

In general, an MS always requests a segment over local Wi-Fi connectivity, if any. If no MS that holds the segment is available, it will request the segment via 3G/4G. After the reception, the MS caches video segments at its own repository; from then on, any MS nearby can share the segments via the local Wi-Fi connectivity. This sharing is easily facilitated in the NDN framework.

We substantiate the sharing concept in NDN on the Android platform as follows. MSs set up a CCNx overlay network on top of UDP/IP sockets. One MS, say the master, offers tethering service to the other MS using the Wi-Fi Direct. The other MS can then obtain video segments directly with the same name from the cache of the master MS. MSs check Wi-Fi link status, and if its link is poor, then they may switch to 3G/4G links.

To achieve the 3G/4G offloading in wireless NDN, every time an MS generates an interest, the CCNx will check whether there is any other MSs nearby. Using the Android SDK we can check whether there is already a Wi-Fi connectivity. Then we send the interest with the same segment name via the IP interface of the local connectivity.

Chapter 5

Details of Multi Interface

5.1 NDN-Femtocell for Edge Caching

We deploy caching functionality in mobile edge, i.e., eNodeB and EPC of LTE on the commercial LTE Femtocell devices offered by Multi CT, and import CCNx into its core network emulator to realize the caching of popular ongoing content and to facilitate the content delivery to the mobile users attached to the cell without going through the core network to the remote source.

The LTE Femtocell offered by MCT has two parts: LTE eNodeB wireless AP, and the CNE which is emulating the EPC functions (S-GW, P-GW and MME) and thus realize the small cells workable as a standalone access point.

The whole structure is all IP-based, and thus provide us opportunity to import CCNx stack. Once the Femtocell is connected to a router with DHCP enabled and with public IP, the mobile client attached to the Femtocell will be assigned with a subnet IP assigned by the router. SDK we can check whether there is already a Wi-Fi connectivity. Then we send the interest with the same segment name via the IP interface of the local connectivity.

5.2 Multi-Interface in Linux

By default, Linux do not allow multi-interfaces to function. Thus, if a Linux machine has multiple network interfaces, we need to enable them by setting routing tables and policy routing. A routing table can contain an arbitrary number of routes, the selection of route is classically made according to the destination address of the packet. In addition to the two commonly used routing tables (the local and main routing tables), the kernel can support up to 252 additional routing tables which can be added in the file /etc/iproute2/rt_tables.

Linux also provides more flexible routing selection based on the Type of Service, scope, output interface. Linux supports policy based routing using the multiple routing table capability and a routing policy database. This database contains routing rules used by the kernel. Using policy based routing, the source address, the ToS flags, the interface name and an "fwmark" (a mark carried through added in the data structure representing the packet) can be used as route selectors. Policy based routing can be used in addition to Linux packet filtering capabilities, e.g., provided by the "iptables" tool. In a multiple interfaces context, this tool can be used to mark the packets, i.e., assign a number to fwmark, in order to select the routing rule according to the type of traffic. This mark can be assigned according to parameters like protocol, source and/or destination addresses, and port number and so on. Such a routing management framework allows to deal with complex situation such as address space overlapping. In this situation, the administrator can use packet marking and policy based routing to select the correct interface.

5.3 Multi-interface in Android

Android is based on a Linux kernel and, in many situations, behaves like a Linux device as described in previous section. As per Linux, Android can manage multiple routing tables and rely on policy based routing associated with packet filtering capabilities. Such a framework can be used to solve complex routing issue brought by multiple interfaces terminals, e.g. address space overlapping.

The Android reference documentation describes the android.net package that applications can use to request the first hop to a specified destination address via a specified network interface (LTE or Wi-Fi). Applications also can ask for permission to start using a network feature. The Connectivity Manager monitors changes in network connectivity and attempts to failover to another network if connectivity to an active network is lost. When there are changes in network connectivity, applications are notified. Applications are also able to ask for information about all network interfaces, including their availability, type and other information. We implement our usage of the android.net package to enable multi-interface communication over Android.

Chapter 6

Implementation and Evaluation

We implemented a wireless NDN test-bed based on Juni's JPW-8000 WiMAX BS. CCNx (version 0.6.1) is successfully ported into the ASN-GW core part, and will help WiMAX BS work in NDN way for caching content effectively and serve for mobile devices. The MCT LTE Femtocell is provided by MCT Company. The base station devices of WiMAX and LTE Femotocell are shown in Figure 6.1. We use laptop based on LTE USB dongle for the LTE service. The Figure 6.2 shows the laptop with LTE USB dongle. We implemented the application on Android 4.1 environment. The mobile client we used is chosen from the HTC EVO 4G+ phone and it is currently the only phone that supports the WiMAX. We also chose the Samsung Galaxy S3 LTE E210K (KT) for the LTE Femtocell support. The original video source for experiment is "Gangnam style" , 800x450, with average bit rate of 773Kbits/s. The Twin is set to 5 seconds. The screenshot of AMVS-NDN is shown in Figure 6.3.

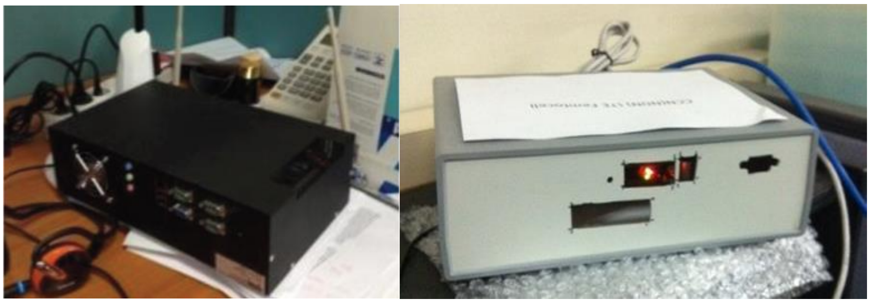
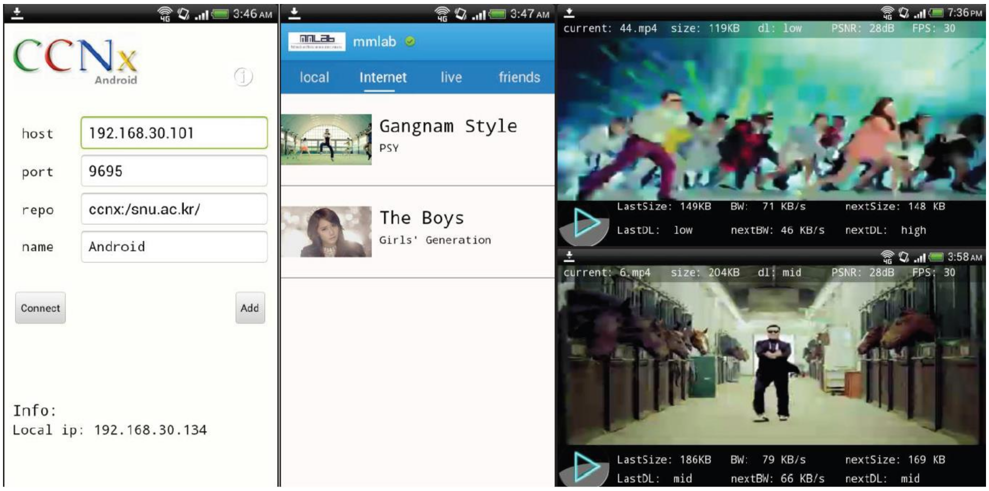


Figure 6.1 Juni JPW8000 WiMAX and MCT LTE Femtocell

Figure 6.2 LTE USB dongle



host 192.168.30.101

port 9695

repo ccnx:/snu.ac.kr/

name Android

Info:
Local ip: 192.168.30.134

Gangnam Style
PSY

The Boys
Girls' Generation

current: 44.mp4 size: 119KB dl: low PSNR: 28dB FPS: 30

LastSize: 149KB BW: 71 KB/s nextSize: 148 KB
LastDL: low nextBW: 46 KB/s nextDL: high

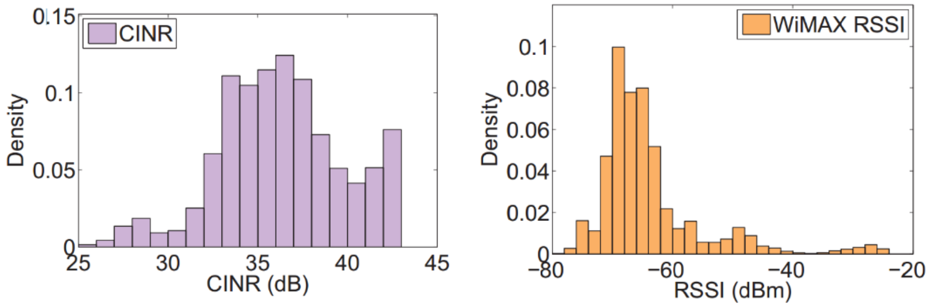
current: 6.mp4 size: 204KB dl: mid PSNR: 28dB FPS: 30

LastSize: 186KB BW: 79 KB/s nextSize: 169 KB
LastDL: mid nextBW: 66 KB/s nextDL: mid

Figure 1.3 Screenshot of AMVS-NDN application

6.1 Testbed Environment

At first, we need to obtain the testing environment of the WiMAX connection. For this purpose, we keep downloading a big video via AMVS-NDN by walking around the floor and capturing the WiMAX signal quality in terms of CINR (Carrier to Interference-plus-Noise Ratio) and RSSI (Received signal strength indication) for the WiMAX environment. Figure 6.4 shows their values measured in the WiMAX environment. In detail, the CINR is always round between 30dB and 45dB, and RSSI is always round between -80dBm and -60dBm, which are under normal cases.



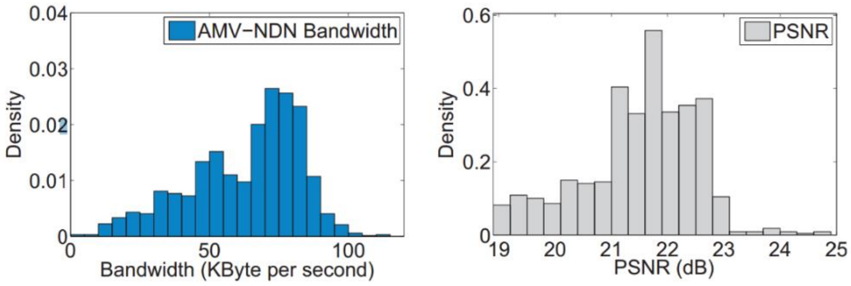
(a) Distribution density - CINR (b) Distribution density - RSSI

Figure 6.4 Experiment results of WiMAX link quality

6.2 AMVS-NDN Evaluation

Based on the practical measurement of the WiMAX testbed environment, we define three levels of video quality profile “low”, “mid” and “high” from the original video. “low” is with low resolution of 320x180 and also low image compress quality, while the bit rate is normally around 160Kbps; “mid” is with mid resolution of 480x270 and also mid image compress quality, while the bit rate is normally around 310Kbps; “high” is with high resolution of 640x360 and also high image compress quality, while the bit rate is normally around 500Kbps. Then we evaluate the AMVS-NDN performance via WiMAX by running it while moving in the floor, and repeat each experiment for 5 times. We further write a logging program to capture the practical CCNx transmission bandwidth. We use PSNR (Peak Signal-to-Noise Ratio) as a metric to evaluate the video quality, which is dynamically shown in the application. PSNR is actually based on a comparison of original and obtained videos, hence we store the captured video at the phone's local storage every one second.

Figure 6.5 shows that the available bandwidth is always between 20KBytes/s and 100KBytes/s, and normally the PSNR of the video is around 19dB to 23dB. Since the signal quality is good in most cases, meaning that video segments with mid and high resolution are mostly displayed, the PSNR falls down to the range of 21dB to 23dB. Note that the PSNR of video display is not only depending on the transmission quality but the video image quality itself can also make a significant impact on the PSNR value.

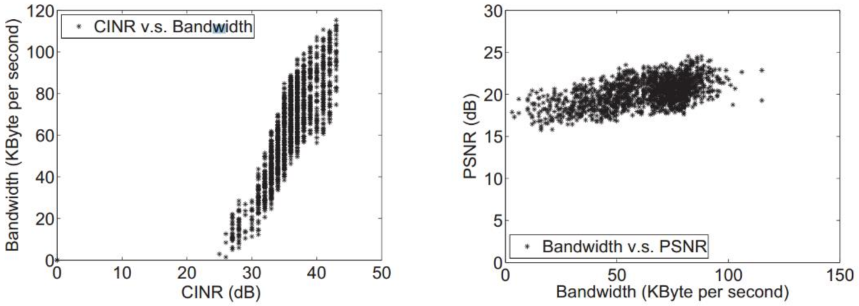


(a) Practical Bandwidth (b) Practical PSNR

Figure 6.5 Experiment result of AMVS-NDN bandwidth and PSNR

In order to show the co-relationship between the link quality and the practical download bandwidth as well as that between the bandwidth and the video quality (PSNR), we plot the scatter figures based on the captured trace. From Figure 6.6(a), we show that there is approximately a linear relation between the CINR and the bandwidth. When the link has the CINR around 35dB to 40dB, the bandwidth can be as high as 80KBytes/s or even 100KBytes/s, but when the CINR drops below 30dB, the bandwidth also falls to around 20KBytes/s to 40KBytes/s. Not only in WiMAX but also in LTE, different link quality will induce different low-layer coding schemes and thus the bandwidth will be changed accordingly.

From Figure 6.6 (b), we also show approximately a linear relation between the bandwidth and the PSNR. When the bandwidth is around 60KBytes/s to 90KBytes/s, the PSNR is mostly clustered around 20dB to 24dB. Thus, in general, higher bandwidth always induces better video quality by PSNR, because of the PSNR is not only decided by the link bandwidth but also the image dynamics. For instance, in some parts, the video contains large portion of a single background color, so even if we deliver the video segment with low resolution, the PSNR can be also very high.



(a) A scatter figure of practical bandwidth to CINR

(b) A scatter figure of PSNR to practical bandwidth

Figure 6.6 Relationships among CINR, bandwidth, and PSNR

6.3 AMVS-NDN Streaming and Sharing

We install our streaming application on one phone (phone A), to obtain the video streams under the WiMAX network, and use another phone connecting to phone A to obtain the viewed video segments with Wi-Fi Direct (or Tethering). During this experiment, we get the bandwidth and PSNR for each phone.

The measured bandwidth of phones A and B are shown in Figure 6.7. The areas with green and blue colors indicate the connectivity via WiMAX link and Wi-Fi Direct (or Tethering), respectively. Note that the bandwidth curves are quite similar to the expected illustration in Figure 3.2. We take the experiment for 4 minutes, and the phone A adaptively obtains the video streaming depending on link quality via the WiMAX connection. At the beginning, Phone B is served by phone A, by obtaining segments that phone A has already obtained from the server via the BS. However, while we move phone B towards the BS (far away from phone A), the link quality of Wi-Fi between two phones drops dramatically and the link quality of WiMAX at phone B becomes better. Then we turn Phone B to connect to the WiMAX BS and continue the transmission. Note that the whole procedure can be automatically done if we obtain the root permission and recompile the source code of the system. However our work is constrained due to the limited support of the HTC phone.

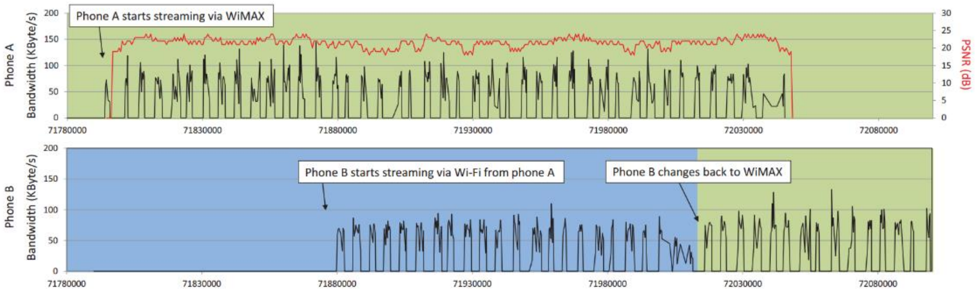
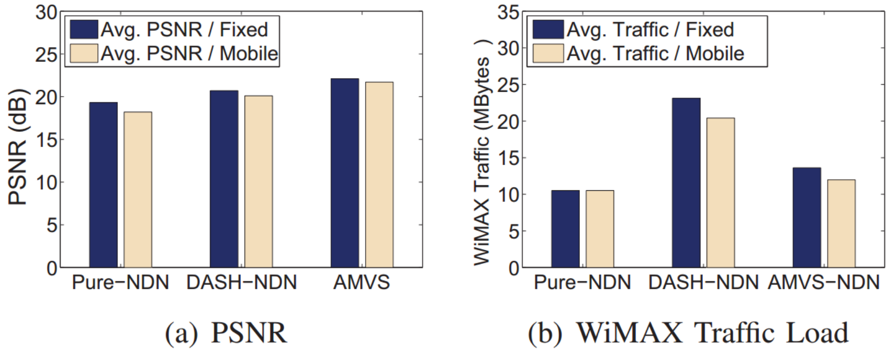


Figure 6.7 Experiment results of AMVS-NDN regarding the adaptive streaming of phone A and video sharing from phone A to phone B

6.4 Comparison with Pure-NDN and DASH-NDN

We compare AMVS-NDN with Pure-NDN and DASHNDN (similar to DASH but augmented for NDN). Pure NDN is constant bit rate (CBR) streaming in NDN without adaptively while DASH-NDN is adaptive streaming in NDN but without phone-to-phone sharing. We reuse the test scenario of the last experiment, that is, phone A is static but phone B moves to the WiMAX BS and switches from the phone-to-phone to the WiMAX connection for better video quality. We average 5 runs as shown in Figure 6.8. In Figure 6.8 (a), in most cases PSNR in the mobile scenario is a little lower than that in the static scenario. Because CBR streaming (Pure-NDN) has to choose a poor quality due to the link dynamics during the whole procedure, the average PSNR is 19.3dB. DASH-NDN supports adaptive streaming, so the average PSNR is a little higher, 21.3dB. However, phone A can work adaptively but phone B sometimes can only obtain a poor quality via the WiMAX link due to the distance in the beginning. AMVS-NDN has the average PSNR of 22.7dB, phone B with a poor WiMAX link quality can easily obtain mid or even high from another phone via Wi-Fi. When phone B changes to WiMAX for better video quality, DASH-NDN and AMVS-NDN may achieve similar performance. Figure 6.8(b) shows the reduction of WiMAX traffic. Pure NDN always obtains video segment with low quality via the WiMAX, so the traffic volume is smaller than DASH-NDN while DASH-NDN always tries to obtain the segment with higher quality via the WiMAX connection depending on the link quality. AMVS-NDN sometimes utilizes the phone-tophone link to share video segments via WiFi. Conclusively, AMVS-NDN outperforms other designs due to its adaptability and sharing, it usually achieves higher video quality (PSNR) but with relatively less cellular traffic.



(a) PSNR

(b) WiMAX Traffic Load

Figure 6.8 A comparison between Pure-NDN, Dash-NDN and AMVS-NDN

6.5 Multi Interface in Linux with LTE access to the Femtocell

The Figure 6.9 shows the test setup of Linux laptop. We use laptop with USB-dongle while also connect to other Wi-Fi resource for multi interface. The caching can be easily deployed into the Femtocell, as the CNE is within the same NAT of the subnet, and we can easily import.

The AMVS-NDN client is deployed into a Linux application based on Java, and we realize the multi interface, laptop-to-laptop sharing and so on, in the similar way to the Android phones. The details of test setup is shown in Figure 6.9.

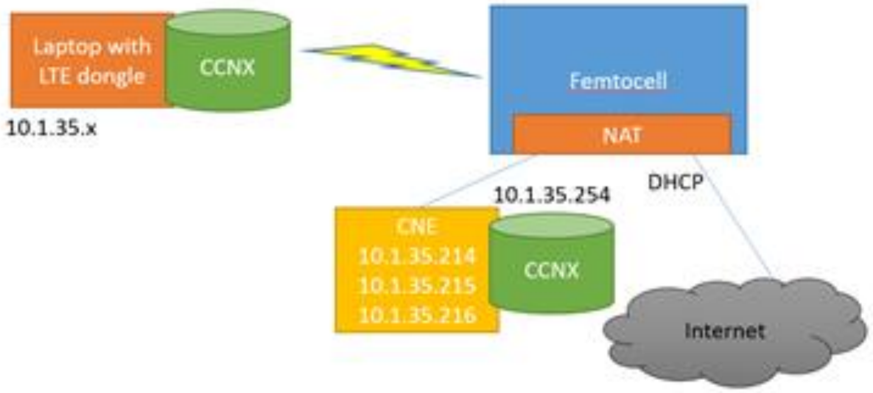


Figure 6.9 Test setup of Linux Laptop

Figure 6.10 shows the experiment for AMVS-NDN with multi interface in Linux system. The laptop with LTE dongle connects to the Femtocell and download the video streaming (“GangNam Style”) through LTE and Wi-Fi interface. Each video streaming will be automatically cached in the CCNx repository while downloading finishes. The smart phone of left can download the same video streaming from laptop’s CCNx repository through Wi-Fi interface, and each streaming will be also cached in repository such like laptop, and the video streaming in repository can be shared with other phone by Wi-Fi Direct.

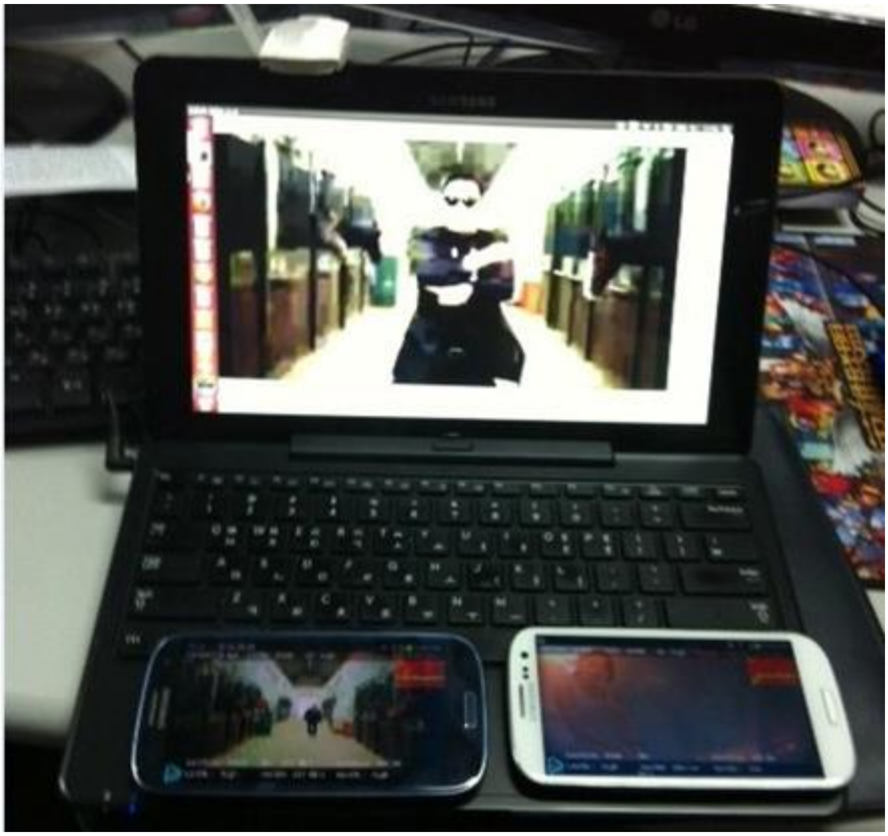


Figure 6.10 Experiment for AMVS-NDN with multi interface in Linux

6.6 Multi Interface in Android with LTE access

The Figure 6.11 shows the test setup for Android system. In this scenario, we test the availability of the whole structure as well as tests on Android phones for the application client, AMVS-NDN, to verify the performance on phones. The details of test setup is shown in Figure 6.11.

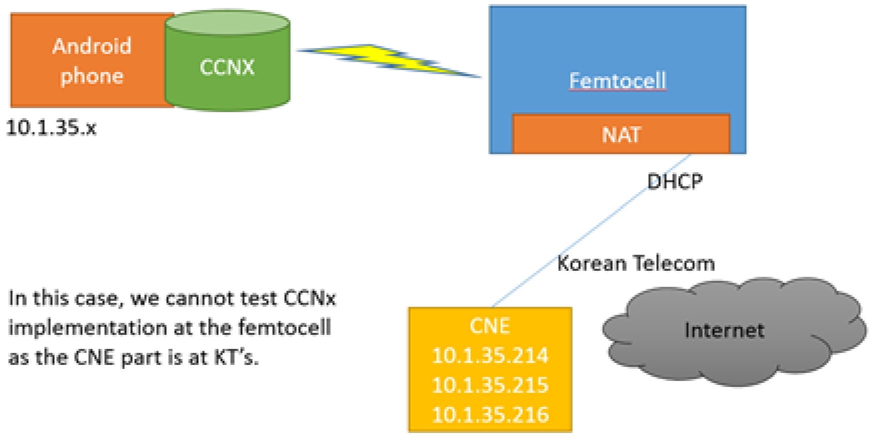


Figure 6.11 Test setup of Android System

We use three Android smartphone, Samsung Galaxy S3, to do the experiment to test the functions we've realized. We name the three Android devices as A, B and C. Test video is stored in CCNx repo in a server. And all the video transmissions are based on CCNx.

For the experiment, first of all, device A connects to LTE Femtocell and WiFi, through which, it can get access to the server, where we stored video for test. LTE and WiFi are working together, called multi-interface. Because of the multi-interface, we can separate the traffic into two path. The transmission policy is simply set like this: one link is responsible for download the odd video tasks, another link is responsible for download the even video tasks. As a result, the transmission rate is significantly improved.

Device B is connected to device A through Wi-Fi. It' s clear that while device A is receiving data from the server, through LTE Femtocell, device B can get this video from deivce A through WiFi. At the same time, we keep another interface, WiFi-Direct working, in order to share the video with another device C.

Device C is connected to device B through Wi-Fi-Direct. It can get the video and play it when device B is playing.

So we conduct the whole experiment like this: device A is connected to the LTE Femtocell and Wi-Fi, connected to the server we stored the test video, through which it can get the test video. The two interfaces are able to work at the same time, improving the transmission rate. And it can share this video with device B through Wi-Fi. For device B, Wi-Fi and Wi-Fi Direct can work together. It gets the video from device A through Wi-Fi, and then shares it with device C using Wi-Fi Direct. We only use one interface, Wi-Fi Direct at device C to get the video from device B. Figure 6.12 shows that the three devices can play the video almost at the same time.

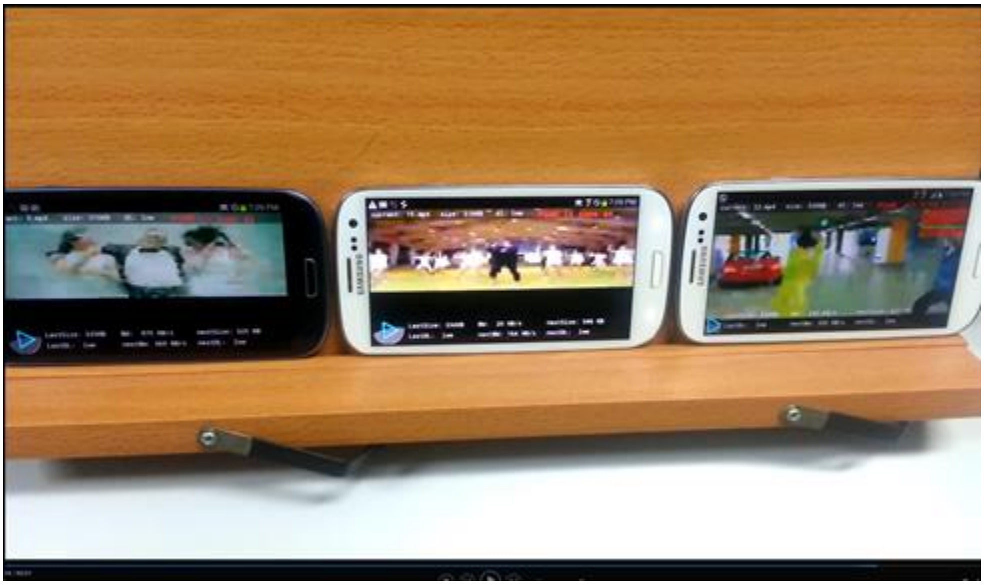


Figure 6.12 Three devices with AMVS-NDN and MIF is working

6.7 Live broadcasting with CCNx and Wi-Fi Direct

In this experiment, we use two Galaxy S3 smart phones to realize the live broadcasting with CCNx and Wi-Fi Direct. The library "VLC" provide the main framework.

Based on the Wi-Fi Direct connection scenario, one phone should be set as group owner, and another phone is set as client. While choosing the device type, 2 phones will try to search each other in the same channel with Wi-Fi Direct. When phone successfully finds another, the CCNx service will automatically start and the live streaming can be used.

The Figure 6.13 shows the situation of live broadcasting experiment. The phone near us records the objects on the desktop, and the phone far from us play the live streaming which is recorded from another phone.

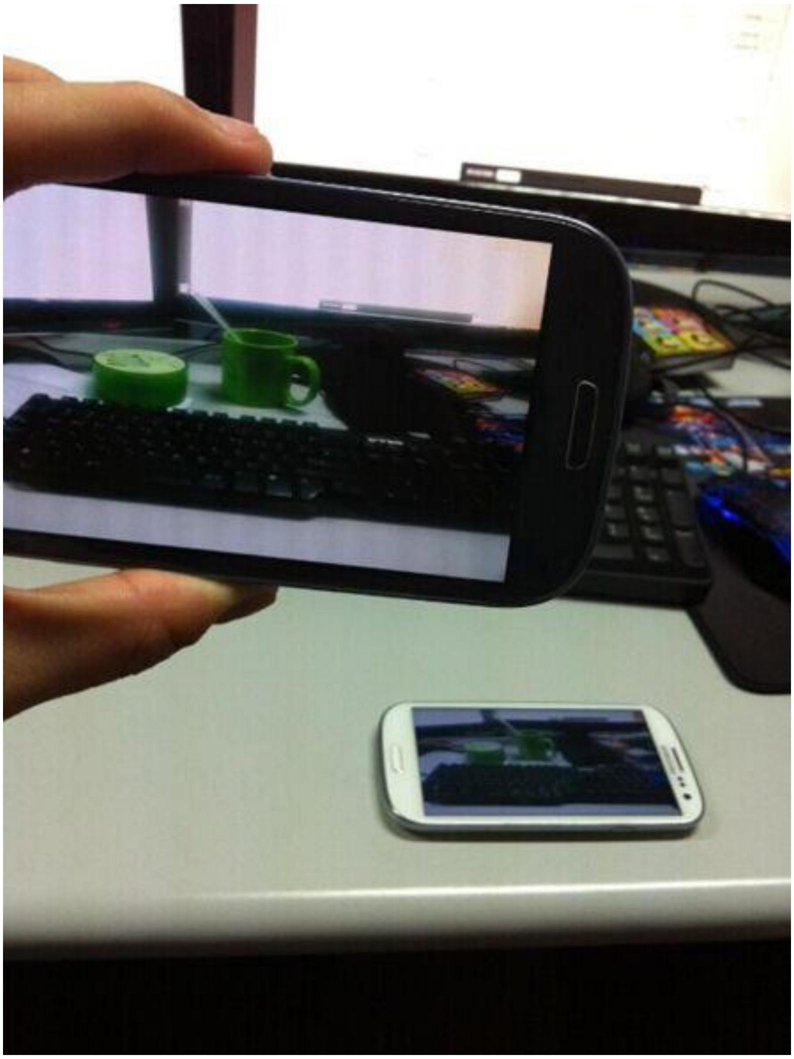


Figure 6.13 Experiments for live broadcasting with CCNx and Wi-Fi Direct

Chapter 7

Conclusion

In this thesis, we discuss the impact of network on the performance of adaptive video streaming in wireless mobile environment, and design an adaptive mobile video streaming with offloading and sharing, AMVS-NDN, in the wireless NDN architecture, along with the functionality for sharing among mobile users by local Wi-Fi connectivity. We also discuss the details of Multi-Interface technology via CCNx. We implement the AMVS-NDN with CCNx, and perform experiments in a real testbed consisting of a WiMAX BS and two phones. It is proved that AMVS-NDN outperforms pure streaming via NDN, and DASH via NDN in terms of the average video quality (PSNR) and traffic load reduction. We also implement AMVS-NDN with multi-interface in Linux and Android system, caching and sharing application is realized such as live broadcasting in CCNx.

Our current work still has some space for improvement: a) we haven't took into account energy consumption yet. However, in some cases total energy consumption may be reduced due to high energy efficiency of Wi-Fi. b) The benefit of Wi-Fi sharing may be not so high in practical when various videos are requested for a short interval. In this case, we encourage to learn from other domains to improve the sharing probability, e.g., social influence from social network services. c) During Wi-Fi sharing the receiver can obtain only video segments that the sender has in its local repository, and thus it may be more flexible if the receiver can request more “extra” segments via the cellular link concurrently for better video quality. d) The segment interval (currently, 5 seconds) should be also adaptive to the link quality; in the future we will carry out more tests to find an optimal interval for various link conditions and mobility scenarios. e) Multi-interface is only used by simple and basic transmission scenarios, various efficient scenarios should be implemented.

References

[1] CISCO, “Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update , 2010-2015,” CISCO, Tech. Rep., 2011.

[2] Project CCNx, “http://www.ccnx.org, ” Sep., 2009.

[3] V. Jacobson, D. Smetters, J. Thornton, M. Plass, N. Briggs, and R. Braynard, “Networking named content,” ACM CoNEXT,2009.

[4] D. Kulinski, J. Burke, “NDNVideo: Random-access Live and Prerecorded Streaming using NDN,” Technical Report NDN-0007, 2012.

[5] H. Yoon, J. Kim, T. Feiselia, H. Robert, “On-demand Video Streaming in Mobile Opportunistic Networks,” IEEE PERCOM, 2008.

[6] Z. Zhenkai, W. Sen Y. Xu, V. Jacobson and Z. Lixia, “ACT: Audio Conference Tool Over Named Data Networking,” ICN, 2011.

[7] A. Zambelli, “IIS Smooth Streaming Overview,” Tech. Rep., 2009.

[8] S. Akhshabi, A. C. Begen, and C. Dovrolis, “An Experimental Evaluation of Rate-Adaptation Algorithms in Adaptive Streaming over HTTP,” MMSys, 2011 [9] V. Singh and I. D. D. Curcio, “Rate Adaptation for Conversational 3G Video,” IEEE INFOCOM Workshop, 2009

[10] K. Tappayuthpijarn, G. Liebl, T. Stockhammer, and E. Steinbach, “Adaptive video streaming over a mobile network with TCP-Friendly Rate Control,” ACM IWCMC, 2009

[11] E. Piri, M. Uitto, J. Vehkaper, and T. Sutinen, “Dynamic Cross-layer Adaptation of Scalable Video in Wireless Networking,” Proceedings of IEEE GLOBECOM, 2010

[12] X. Wang, M. Chen, B. Pan, T. Kwon, L. T. Yang, V. C. M. Leung, “AMES-Cloud: A Framework of Adaptive Mobile Video Streamingand Efficient Social Video Sharing in the Clouds,” IEEE Transaction of Multimedia, 2013.

[13] K. Doppler, M. Rinne, C. Wijting, C. Ribeiro and K. Hugl, “ Devicetodevice communication as an underlay to LTE-advanced networks,” IEEE Communications Magazine, vol.47, issue.12, pp.42-49, 2009.

[14] P. Kyasanur, and N. Vaidya, “Routing and Interface Assignment in Multi-Channel Mutil-Interface Wireless Networks ” IEEE Wireless Communications and networking Conference, 2005.

[15] P. Kyasanur, and N. Vaidya, “Routing and Link-layer Protocols for Multi-Channel Multi-Interface Ad Hoc Wireless Networks” ACM Mobile Computing and Communications Review, vol.10, Num 1, 2013.

[16] S. Lederer, C. Mueller, B. Rainer, C. Timmerer and H. Hellwagner, “Adaptive Streaming over Content Centric Networks in Mobile Networks using Multiple Links” IEEE International Conference on Communications, 2013

초록

최근의 모바일 트래픽(특히 비디오 트래픽)의 폭발적인 증가는 모바일 네트워크 사업자들에게 큰 부담이 되고 있다.비디오 스트리밍 서비스가 모바일 사용자에게 주요한 서비스가 됨에 따라 모바일 트래픽이 네트워크망의 대역폭을 초과하는 경우가 잦아지고 있다. 이 문제를 해결하기 위해 우리는 “이름 주소 기반 네트워킹(NDN)” 이라는 차세대 인터넷 아키텍쳐를 이용했다. NDN 은 혁신적인 네트워크 아키텍처로서 인터넷의 차기 모델로서 평가되고 있다.

우리는 NDN 아키텍처 상에서 다중 무선 인터페이스(4G LTE, Wi-Fi)를 활용한 어댑티브 비디오 스트리밍/셰어링 기술을 개발하고 이를 적용한 프레임워크 (AMVS-NDN)를 디자인하고 구현하였다. NDN 의 장점을 보여주기 위해 AMVS-NDN 은 두가지의 기능을 제공한다. 1) 기본적으로 모바일 스테이션(MS)은 상황에 따라 4G LTE 나 Wi-Fi 중 하나를 선택해 사용한다. 이에 더해, 다중 인터페이스 기술을 사용하여 4G LTE 와 Wi-Fi 를 동시에 사용할 수 있다. 2) MS 들은 로컬 Wi-Fi direct 를 활용하여 콘텐츠를 공유할 수 있다. 우리는 NDN 과 다중 인터페이스를 이용하여 AMVS-NDN 을 구성했다. 테스트는 WiMAX 베이스 스테이션, LTE Femtocell, Android 디바이스로 구성된 실제 테스트베드에서 수행되었다. 모바일 링크의 상태가 시간에 따라 변하는 상황의 테스트를 통해, 우리는 AMVS-NDN 이 다른 솔루션보다 적은 셀룰러 트래픽으로 높은 비디오 품질을 달성하는 것을 보여주었다. 더욱이, 다중 인터페이스 기술을 적용시 최고의 비디오 품질을 얻을 수 있었다.

키 워드 : 이름 주소 기반 네트워킹, 어댑티브 비디오 스트리밍,
오프로딩과 셰어링, 다중 인터페이스.

학 번 : 2013-22515