# AMVS-NDN: Adaptive Mobile Video Streaming and Sharing in Wireless Named Data Networking

Bing Han, Xiaofei Wang, Nakjung Choi*, Ted "Taekyoung" Kwon, and Yanghee Choi
School of Computer Science and Engineering, Seoul National University, Seoul, Korea
Email: {binghan, dobby}@mmlab.snu.ac.kr, {tkkwon, yhchoi}@snu.ac.kr
*Bell-Labs, Alcatel-Lucent, Seoul, Korea
Email: nakjung.choi@alcatel-lucent.com

***Abstract*—Recently, mobile traffic (especially video traffic) explosion becomes a 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). In this paper, we design and implement a framework of adaptive mobile video streaming and sharing in the NDN architecture (AMVS-NDN) considering that most of mobile stations have multiple wireless interfaces (e.g., 3G and WiFi). To demonstrate the benefit of NDN, AMVS-NDN has two key functionalities: (1) a mobile station (MS) seeks to use either 3G/4G or WiFi links opportunistically, and (2) MSs can share content directly by exploiting local WiFi connectivities. We implement AMVS-NDN over CCNx, and perform tests in a real testbed consisting of a WiMAX base station 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.**

***Index Terms*—Named Data Networking, Adaptive Video Streaming, Mobile Networks, Offloading and Sharing.**

## I. Introduction

Recently, most of mobile network operators are facing a serious challenge due to mobile data (especially video 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] can be an attractive solution, adapting the network architecture to the current network usage pattern, i.e., data dissemination. 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 also 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 example, it will be fairly efficient and hence necessary to satisfy requirements, e.g., latency, of video delivery to cache popular videos at clustering points, e.g., base station (BS) in WiMAX and access point (AP) in WiFi. 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, WiFi, 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 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 WiFi, a MS receives video streams via a cellular link and also opportunistically accesses local WiFi 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].

In this work, by adding functionalities for adaptive video streaming and sharing among MSs with local WiFi, we realized adaptive mobile video streaming and sharing in the NDN architecture, termed AMVS-NDN. We implemented AMVS-NDN within CCNx, and conducted experiments in a real test-bed consisting of a WiMAX BS and two MSs equipped with WiFi. 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.

The rest of the paper is organized as follows. We first introduce related work in Section II, and explain details of the AMVS-NDN framework in Section III. Then, the video sharing will be further discussed in Section IV. We evaluate the prototype implementation in Section V, and finally conclude this paper in Section VI.

## II. Related Work

### A. 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.

### B. 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.

### C. 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 became 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 and WiFi, in the NDN architecture [5].

## III. AMVS-NDN Framework

### A. 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 WiFi link between the two MSs is available (with high bandwidth). Then MS $B$ switches to the WiFi link to download the video data from MS $A$ directly. MS $B$ can connect to MS $A$ by the WiFi 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 WiFi direct connectivity. Then MS $B$ will receive the video data via the BS, and MS $A$ will receive the data from MS $B$.

### B. 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 Worldcup 2010. Then the name of the $23^{rd}$ video segment with low video quality can be like, **/fifa.com/video/worldcup2010/final/_low/_023**. The name structure can be directly found out from the search engine or figured out from the metadata file.

### *C. 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 $T_{win}$ seconds, the procedure of throughput estimation and bit rate adjustment is shown in Algorithm 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.

**Algorithm 1** Bandwidth Estimation and Quality Adjustment

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

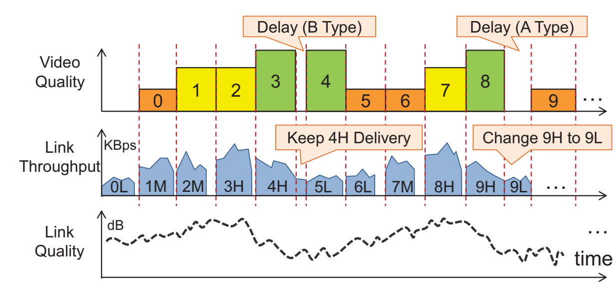



Fig. 1. The link quality, link throughput, and playback delay in AMVS-NDN are illustrated over time.

### *D. Dealing with Delays*

In the current CCNx 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 Fig. 1, 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.

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 $T_{win}$: **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 disruption in the playback. We interrupt the current transmissions with I/O error exception of NDNinputstream in CCNx API and transmit the interest for the low quality segment immediately. At the top of Fig.1, 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 Fig.1, 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.

## IV. 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 WiFi 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 WiFi 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 WiFi Direct. The other MS can then obtain video segments directly with the same name from the cache of the master MS. MSs check WiFi 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 WiFi connectivity. Then we send the interest with the same segment name via the IP interface of the local connectivity.

## V. 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. We implement 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. The original video source for experiment is "Gangnam style", 800x450, with average bit rate of 773Kbits/s. The $T_{win}$ is set to 5 seconds. The screenshot of AMVS-NDN is shown in Fig. 2.

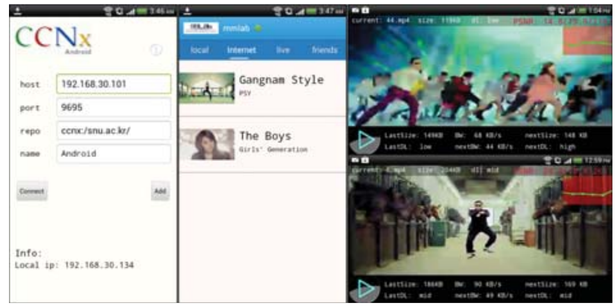
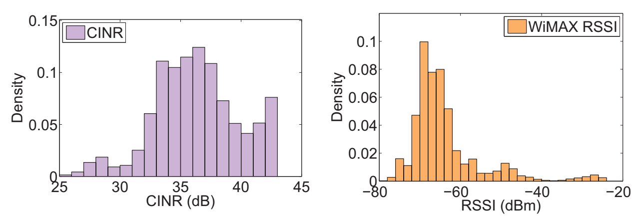



Fig. 2. Screenshot of AMVS-NDN application

### A. 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. Fig. 3 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

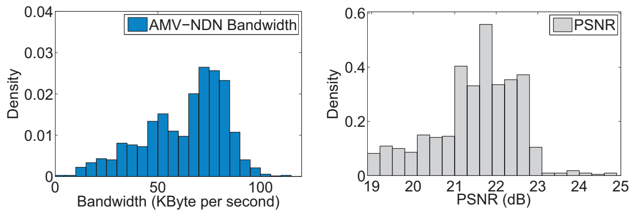
Fig. 3. Experiment results of WiMAX link quality

### B. 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.

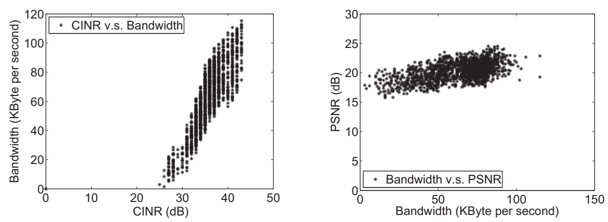
Then we evaluate the AMVS-NDN performance via WiMAX by running it while moving in the floor, and repeat each experiment for 5 times. We further write a logging program to capture the practical CCNx transmission bandwidth.

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.




(a) Practical Bandwidth (b) Practical PSNR

Fig. 4. Experiment results of AMVS-NDN bandwidth and PSNR




(a) A scatter figure of practical bandwidth to CINR (b) A scatter figure of PSNR to practical bandwidth

Fig. 5. Relationships among CINR, bandwidth, and PSNR

Fig. 4 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.

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 Fig. 5(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 Fig. 5(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.

### *C. 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 WiFi 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 Fig. 6. The areas with green and blue colors indicate the connectivities via WiMAX link and WiFi Direct (or Tethering), respectively. Note that the bandwidth curves are quite similar to the expected illustration in Fig. 1. 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 WiFi 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.

### *D. Comparison with Pure-NDN and DASH-NDN*

We compare AMVS-NDN with Pure-NDN and DASH-NDN (similar to DASH but augmented for NDN). Pure-NDN is constant bit rate (CBR) streaming in NDN without adaptivity 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 Fig. 7.

In Fig. 7(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 WiFi. When phone B changes to WiMAX for better video quality, DASH-NDN and AMVS-NDN may achieve similar performance.

Fig. 7(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,

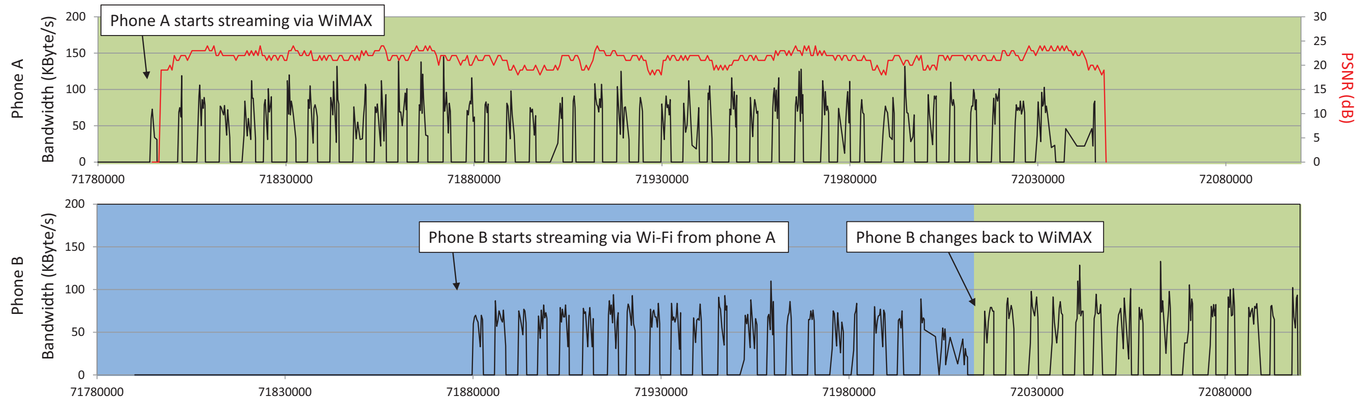
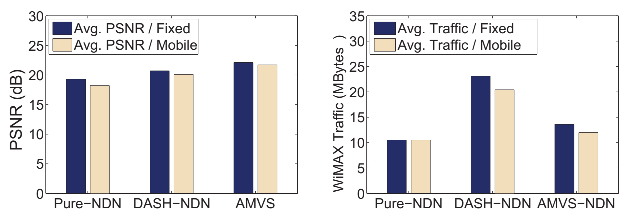



Fig. 6. Experiment results of AMVS-NDN regarding the adaptive streaming of phone A and video sharing from phone A to phone B

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-to-phone 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

Fig. 7. A comparison between Pure-NDN, Dash-NDN and AMVS-NDN

## VI. Conclusion

In this paper, 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 WiFi connectivity. 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.

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 WiFi. b) The benefit of WiFi 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 WiFi 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.

## Acknowledgment

This work was supported by the KCC(Korea Communications Commission), Korea, under the R&D program supervised by the KCA(Korea Communications Agency) (KCA-2012-11-911-05-002), and also was in part supported by Seoul R&BD Program (WR080951),

## 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 Pre-recorded 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, A. 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.IEEE*, 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, "Device-todevice communication as an underlay to LTE-advanced networks," *IEEE Communications Magazine*, vol.47, issue.12, pp.42-49, 2009.