Cooperative Video Streaming on Smartphones

Hülya Seferoğlu†, Lorenzo Keller‡, Blerim Cici†, Anh Le†, Athina Markopoulou†, Christina Fragouli‡

†University of California, Irvine
‡ École Polytechnique Fédérale de Lausanne

Supported by AFOSR MURI, NSF CAREER and by ArmaSuisse W+T
“Unofficial” Motivation

• Typical practical questions after a talk on network coding:
  – “Is it implemented in practice?”
  – “How much benefit does it bring?”
  – “How much is the overhead”?
  – “Can non-network coding approaches do as well?”
  – “I don’t believe this is useful!”

• We decided to implement network coding ...
Cooperative Video Streaming on Smartphones

Scenario

- Several users, within proximity of each other
- Interested in viewing the same video at the same time (e.g. a soccer match or lecture)
- Have Internet connection via cellular or wifi
- Mobile-to-mobile connections possible via bluetooth or wifi.
Cooperative Video Streaming on Smartphones

Examples

- Each individual user may not have high enough rate to stream a high quality video (due to bad, time-varying, or no cellular connection)

- Special case: sharing a video stored on one of the phones
Cooperative Video Streaming on Smartphones

Question

- How to best utilize all resources to provide the best performance?
Cooperative Video Streaming on Smartphones

Key Ingredients

1. Cooperation:
   use cellular links and local links

2. Network coding:
   intra-session, application-layer

3. Broadcast:
   use wireless+NC efficiently

4. Implementation on Androids

This scenario lends itself naturally to network coding.
**Related Work**

- **Smartphones with NC Implementation**
  - Baochun Li’s group: NC implementation in Iphones and Ipad.

- **Wifi devices with NC Implementation**
  - COPE testbed, ...

- **Network Coding-based P2P**
  - C. Gkantsidis, P. R. Rodriguez, "Network coding for large scale content distribution," *in Proc. of Infocom, Miami, FL, March 2005.*
  - Z. Liu, C. Wu, B. Li, S. Zhao, "UUSee: large-scale operational on demand streaming with random network coding," *in Proc. of Infocom, San Diego, CA, March 2010.*
Related Work

- **Network Coding & NUM**

- **Network Coding & NUM & P2P**
MicroCast

Key ingredients - revisited

1. Cooperation:
   use cellular links and local links
2. Network coding:
   intra-session, application-layer
3. Broadcast:
   use wireless+NC efficiently
4. Implementation on Androids
Outline

• NUM formulation

• Implementation and demo on androids

• Work in progress..
NUM Formulation

Cooperation & Unicast

\[
\text{Optimize video rate: } x = \max_x U(x)
\]

\[
\sum_{i \in N} x_{i,j} - x \geq 0, \forall j \in N
\]

\[
g_{i,j} - x_{i,j} \geq 0, \forall i \in N, j \in N - \{i\}
\]

\[
x_{i,j} \leq C_i (1 - p_i), \forall i \in N, j \in N
\]

\[
g_{i,j} \leq C_{i,j} (1 - p_{i,j}) \tau_{i,j}, \forall i \in N, j \in N - \{i\}
\]

\[
\sum_{i \in N} \sum_{j \in N - \{i\}} \tau_{i,j} \leq \gamma
\]

Flow conservation constraints
Capacity constraints
Interference

Downlink 3g/4g

The Source

Smartphones

\[
C_{i,j}, p_{i,j}
\]
NUM Formulation
Cooperation & Broadcast

\[
\begin{align*}
\text{max}_x & \quad U(x) \\
\text{s.t.} & \quad \sum_{j \in J} x_{i,j} - x & \geq 0, \forall j \in N \\
& \quad g_{i,j} - x_{i,j} & \geq 0, \forall i \in N, j \in N - \{i\} \\
& \quad x_{i,j} & \leq C_i (1 - p_i), \forall i \in N, j \in N \\
& \quad g_{i,j} & \leq \sum_{j' \in J} f_{i,j'}, \forall i \in N, j \in N - \{i\} \\
& \quad f_{i,j} & \leq \min_{j' \in J} \{C_{i,j'}\} \prod_{j' \in J} (1 - p_{i,j'}) \tau_{i,j'}, \forall i \in N, j' \in H \\
& \quad \sum_{i \in N} \sum_{j \in H} \tau_{i,j} & \leq \gamma \\
\end{align*}
\]

Optimize video rate: \( x \)

Flow conservation constraints
Downlink and local area capacity constraints
Interference
NUM Formulation
Cooperation & Broadcast & NC

\[
\begin{align*}
\text{max}_{x} \quad & U(x) \\
\text{s.t.} \quad & \sum_{i \in N} x_{i,j} - x \geq 0, \forall j \in N \\
& g_{i,j} - x_{i,j} \geq 0, \forall i \in N, j \in N - \{i\} \\
& x_{i,j} \leq C_i (1 - p_i), \forall i \in N, j \in N \\
& g_{i,j} \leq \sum_{j \in \mathcal{J}} f_{i,j}, \forall i \in N, j \in N - \{i\} \\
& f_{i,j} \leq \min_{j \in \mathcal{J}} \{C_{i,j} (1 - p_{i,j})\} \tau_{i,j}, \forall i \in N, J \in \mathcal{H} \\
& \sum_{i \in N} \sum_{j \in \mathcal{J}} \tau_{i,j} \leq \gamma
\end{align*}
\]

Optimize video rate: \( x \)

Flow conservation constraints
Downlink and local area capacity constraints
Interference
NUM Solution

Rate Control at the Source
\[ x = (U')^{-1}(\sum_{j \in J} \lambda_j) \]

Downlink Rate Control
\[ \max_x \sum_{i \in N} \sum_{j \in N} x_{i,j} (\lambda_j - \eta_{i,j}) \]
\[ \text{s.t. } x_{i,j} \leq C_i (1 - p_i), \forall i \in N, j \in N \]

Queue Update at the Source
\[ \lambda_j(t + 1) = \{\lambda_j(t) + \beta_t [x(t) - \sum_{i \in N} x_{i,j}(t)]\}_{+}, \forall j \in N \]

Queue Update at Smartphones
\[ \eta_{i,j}(t + 1) = \{\eta_{i,j}(t) + \beta_t [x_{i,j}(t) - g_{i,j}(t)]\}_{+}, \forall i \in N, j \in N \]

Local Area Rate Control/Scheduling

Cooperation & Broadcast & NC
\[ \max_{\tau} \sum_{i \in N} \sum_{j \in J} \tau_{i,j} \left( \sum_{j \in J} \eta_{i,j} \min\{C_{i,j} (1 - p_{i,j})\} \right) \]
\[ \text{s.t. } \sum_{i \in N} \sum_{j \in J} \tau_{i,j} \leq \gamma \]

Cooperation & Broadcast & No-NC
\[ \max_{\tau} \sum_{i \in N} \sum_{j \in J} \tau_{i,j} \left( \sum_{j \in J} \eta_{i,j} \min\{C_{i,j}\} \prod_{j \in J} (1 - p_{i,j})\right) \]
\[ \text{s.t. } \sum_{i \in N} \sum_{j \in J} \tau_{i,j} \leq \gamma \]

Cooperation & Unicast & NC/No-NC
\[ \max_{\tau} \sum_{i \in N} \sum_{j \in N} \eta_{i,j} C_{i,j} (1 - p_{i,j}) \tau_{i,j} \]
\[ \text{s.t. } \sum_{i \in N} \sum_{j \in N} \tau_{i,j} \leq \gamma \]
Performance Evaluation
Numerical Results: throughput vs # of users

\[ C_i = 1, \ p_i = 0, \ C_{i,j} = 1, \ p_{i,j} = 0 \]

\[ C_i = 1, \ p_i = 0, \ C_{i,j} = 1, \ p_{i,j} = 0.2 \]
From NUM to implementation

○ What do we learn?
  ○ How to determine source video rate?
  ○ Optimal downlink rates?
  ○ Optimal mobile-to-mobile rates
  ○ Which mobile should transmit?
  ○ What information needs to be exchanged?

○ Implementation should mimic the optimal solution

○ Can use the numerical results to check real experiments, understand the effect of various parameters, and for provisioning
Implementation

Hardware

- **Hardware platform**
  - Google Nexus S
    - 1 GHz Cortex A8, 512 MB RAM

- **Network connections**
  - cellular (3G)
  - 802.11b
  - Bluetooth 2.1+EDR
Implementation
Software

- **Software platform**
  - Android 2.3
  - Application written in Java runs on
    - Android (mobile phones, tablets)
    - Java 2 SE (laptops, etc.)
Application Architecture

GUI and Video player

Segment reordering and caching

Requester

Abstraction layer

Download segments from source

Downlink: Wifi/3G

Local Collaboration

Network layer

Local link: Bluetooth/WiFi
Coding: not a challenge

- We used NCUtils
  - Java (and C) library available at: [http://arni.epfl.ch/software/](http://arni.epfl.ch/software/)
  - Multiplication and inverse done with table lookup, addition done with XOR

- Encoding/Decoding:
  - RNC encoding, early decoding
  - could support rates >6Mbps

- Parameters (not optimized)
  - Symbol: 1B (efficient to implement)
  - Generation size 10. ~ 10KB RAM to decode each generation
  - Payload size 1000 bytes
  - Coding coefficient overhead 1%
Multiple Interfaces: a challenge

• Android is optimized for space and battery
  – 3G, WiFi, Bluetooth

• Simultaneously using multiple interfaces:
  – Bluetooth and 3G: ok
  – Bluetooth and wifi: on the same chip
  – 3G and WiFi: one sleeps while the other is active
  – (use WiFi for both downlink and local links: MAC contention)
  – We had to re-implement a network layer to provide routing and datagram communication at the application layer
Broadcast: a major challenge

- Bluetooth: currently not supported
- WiFi: doable but difficult

- WiFi Broadcast: not the best option (base rate, reliability)
- WiFi Pseudo-broadcast on androids (vs. PCs) is challenging:
  - Have to root the phone
  - Driver for overhearing not available on all devices
  - Does not work in ad-hoc mode
  - Sniffer: we had to extend the Android API to support overhearing
HTTP Server

NC Proxy

Base station

Testbed

1 2 3 4

10...
Demo #1: The value of (1) Cooperation and (2) NC
by Anh Le @UCI: Users 1 and 2 download a 1.57MB file

3G alone
3G downlinks, independent pull

Cooperation
3G downlinks, pull-based cooperation no NC

Cooperation+NC
3G downlinks, push+NC cooperation push+NC proxy

HTTP Server
Base station

55 secs, 239 kbps

40 secs, 329 kbps

32 secs, 411 kbps
Demo #2: The Value and Feasibility of (3) Pseudobroadcast
by Blerim Cici @UCI: A downloads a 4.92MB mp3 and sends it to B and C over UDP.

Unicast
- WiFi downlink
- independent pull
- no overhearing

Pseudo-broadcast
- WiFi downlink
- independent pull
- NC+overhearing

56 sec (on avg)
12.2 segments lost at B,
8.48 segments lost at C

40 sec (on avg)
2.5 segments lost at B,
0.5 segments lost at C
Next Steps

• Analysis
  – bridge the gap between the NUM formulation and the practical implementation

• Implementation
  – incorporate WiFi pseudobroadcast with Cooperation+NC
  – make MicroCast publicly available as:
    • Android application
    • PC/laptop application
    • Java library
Thank you.

http://odysseas.calit2.uci.edu/doku.php/public:muri09