Cross-layer Energy Optimization for Dynamic Video Streaming over Wi-Fi

Zhi Zhang, Mehmet Karaca, Farnaz Moradi, Saeed Bastani,
Anders Nilsson Plymoth, Rickard Ljung, Björn Landfeldt

Energy Savings for Video Streaming Using Fountain Coding

Anders Nilsson Plymoth, Zhi Zhang

TelHoc

Lund University, Lund, Sweden
TelHoc AB, Stockholm, Sweden
Sony Mobile AB, Lund, Sweden

LUND UNIVERSITY

Outline

- Introduction
- Related Work
- Background and Research Problem
- Cross-layer Algorithm
- Performance Evaluation
- Conclusion
- Acknowledgment

Introduction 1

- Motivation
  - Mobile video streaming over Wi-Fi: Energy hungry
  - Dynamic Power Saving Mode (DPSM) in Wi-Fi: fixed timeout. Large timeout => low latency, high energy consumption
- Contribution
  - Propose a lightweight distributed cross-layer algorithm
  - RTT of video segments at APP + DPSM Timeout at MAC => next timeout + next video quality

Related Work

- PSM-AW [7]
- Sleep scheduling [8]
- Adaptive PSM [9]
- Traffic shaper and profiler [10]
- Mobility aware PSM [11]
- Proxy server assisted PSM [12]
- Predictive Green Streaming [13]
- Dynamic Listen Interval [14]
- Dynamic sleep/active scheduling [15]

Background and Research Problem

- Dynamic Adaptive Streaming over HTTP (DASH)

$$avgThr = \frac{Totbit}{avgRTT} \Rightarrow \begin{cases} \text{Increase} & l_c = min(l_{c+1}, l_L) \quad avgThr \geq B_{l_{c+1}} \\ \text{Decrease} & l_c = max(l_{c-1}, l_1) \quad avgThr < B_{l_c} \\ \text{Remain} & \text{Otherwise} \end{cases}$$

- DPSM in Wi-Fi

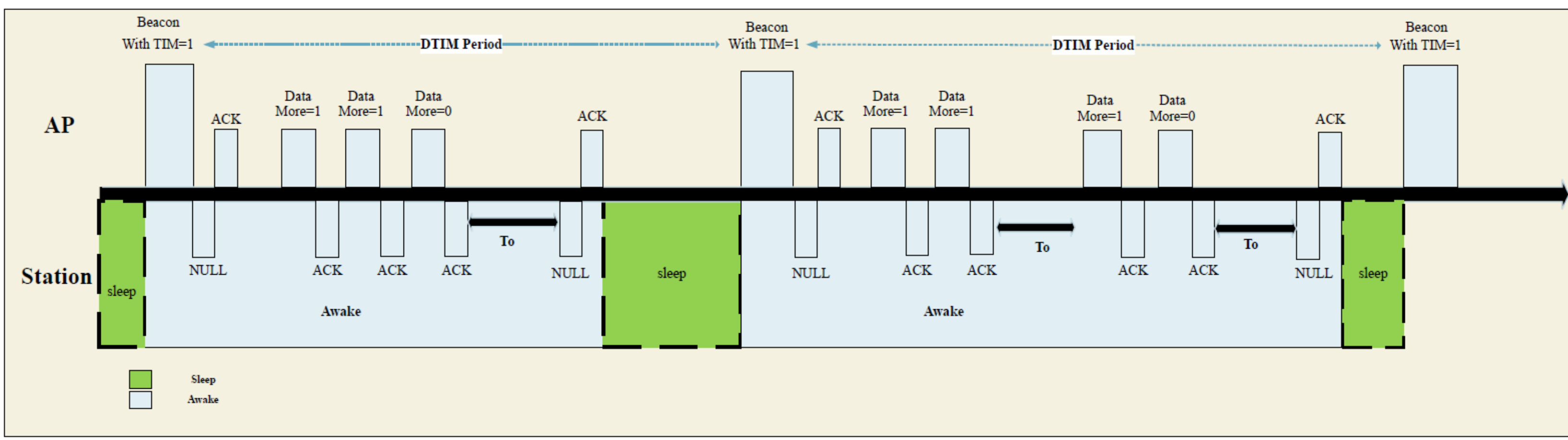


Fig. 1: DPSM with the latest packet

Cross-layer Algorithm

- Fixed Timeout for Fixed Video Quality

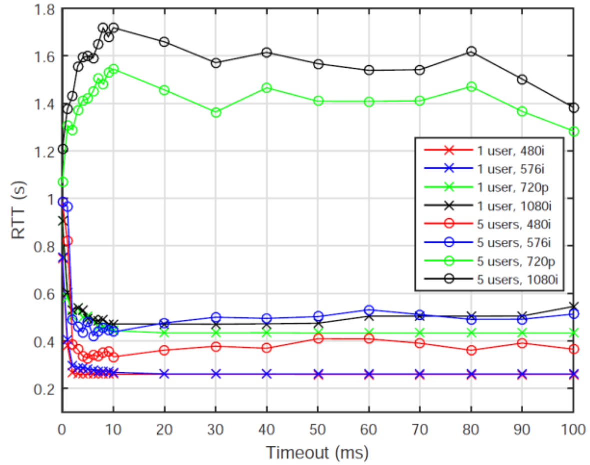


Fig. 2: Average RTT for Different Video Qualities

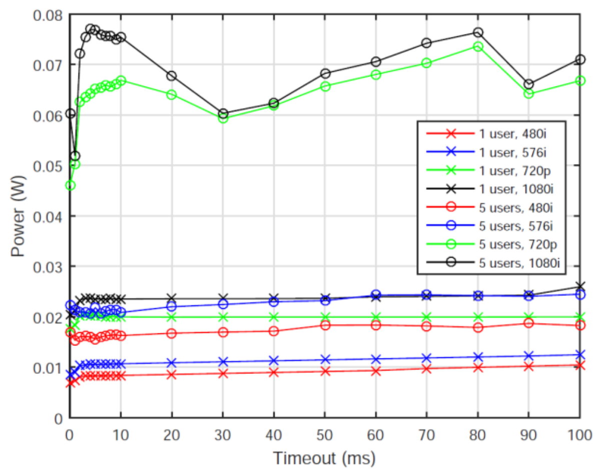


Fig. 3: Average Power Consumption for Different Video Qualities

- In Fig. 2, for a single user, RTT first decreases significantly and becomes almost flat when the timeout increases. For multiple users, if a user increases timeout, there are two contradictory effects: it prompts to reduce its own RTT but in the meanwhile also increases the waiting time for other users. The two effects cause fluctuation of the curves for multiple users.

- In Fig. 3, for a single user, the power consumption increases fast and then grows slightly when the timeout increases. However, for multiple users, the power consumption varies due to the fluctuation of arrived packets caused by the RTT fluctuation.

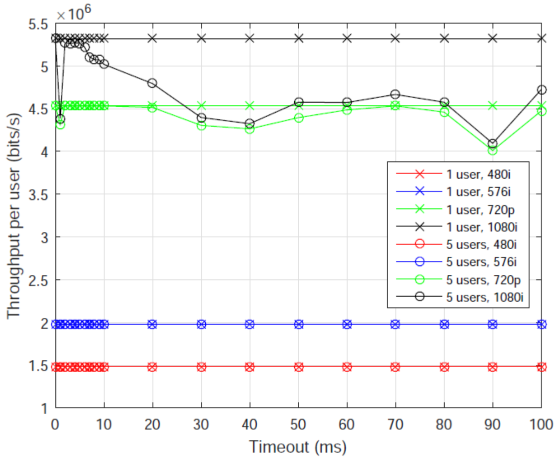


Fig. 4: Average Throughput per User for Different Video Qualities

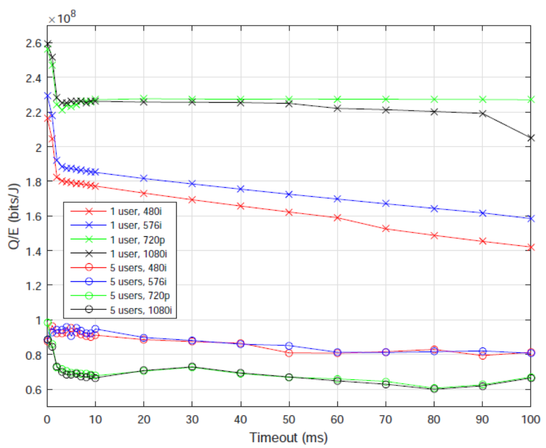


Fig. 5: Average Efficiency for Different Video Qualities

- In Fig. 4, the network can service the requested throughput and video playback is smooth for the single-user case. However, for multiple users with high quality, the network cannot meet the throughput requirement at times, which leads to a playback halt for some users.

- In Fig. 5, for a single user, a higher video quality achieves better efficiency, whereas for multiple users, low qualities are more efficient than high qualities.

- Dynamic Timeout for DASH

Algorithm 1: Dynamic timeout

Given δ_1, δ_2, ϵ, T_o^l, T_o^u. For every new video content, do:

Step 0: Set the initial timeout to its maximum value,

$$to(t) = T_o^u$$

Step 1: Measure RTT over last K packets, denote it by $avgRTT$.

Step 2: Decrease timeout as follow;

$$to^{'}(t) = to(t)/\delta_1$$

If $to^{'}(t) < T_o^l$, then $to(t) = T_o^l$. Otherwise, $to(t) = to^{'}$.

Step 3: Measure RTT over K packets with new timeout, denote it by $newavgRTT$.

Step 4.1: Use $newavgRTT$ to determine video quality using R-DASH.

Step 4.2: **if**

$$|newavgRTT - avgRTT| < \epsilon$$

then set $avgRTT \leftarrow newavgRTT$, and go Step 2 to further decrease the timeout.

else

$$to^{''}(t) = to(t) + \delta_2$$

If $to^{''}(t) > T_o^u$, then $to(t) = T_o^u$. Otherwise, $to(t) = to^{''}(t)$. Set $avgRTT \leftarrow newavgRTT$, Go Step 3.

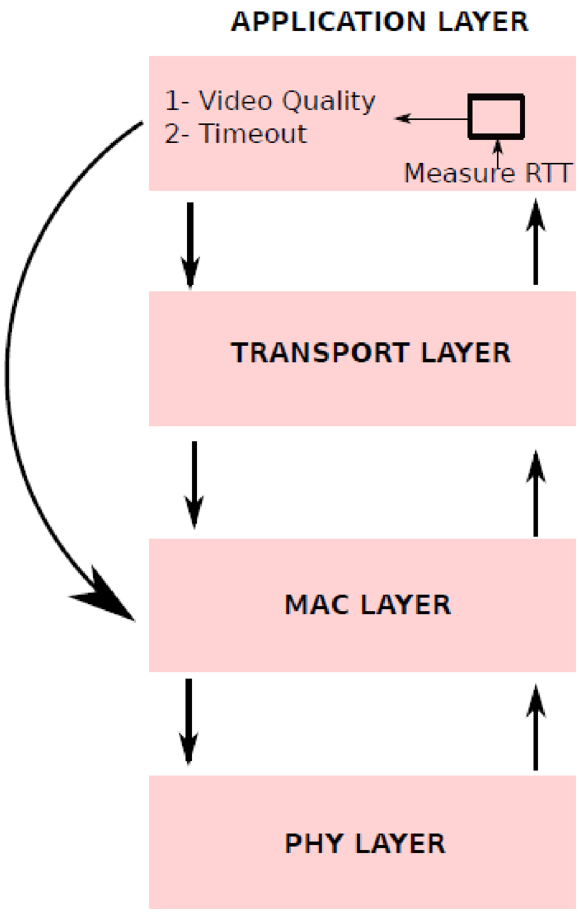


Fig. 6: DPSM with the latest packet

Performance Evaluation

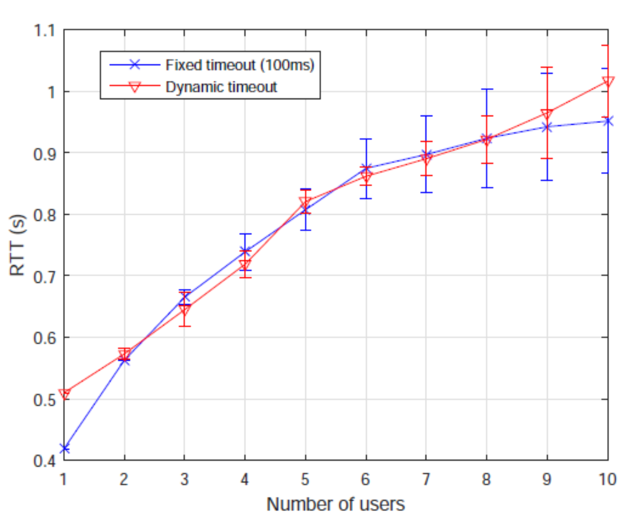


Fig. 7: Average RTT for DASH

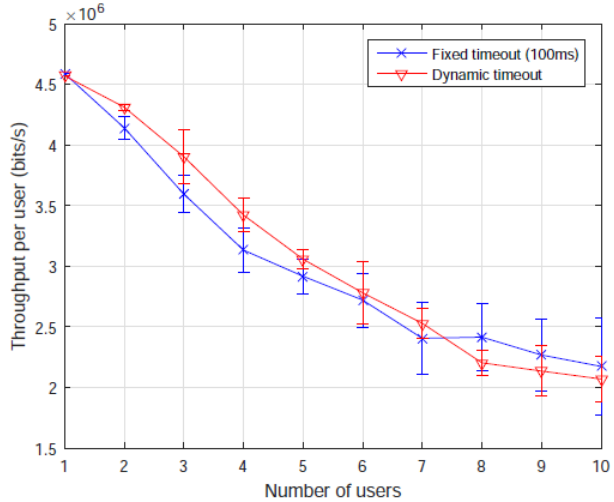


Fig. 9: Average Throughput per User for DASH

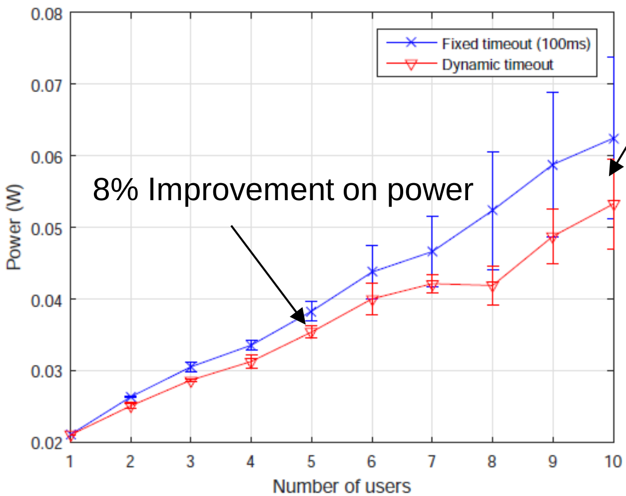


Fig. 8: Average Power Consumption for DASH

15% Improvement on power

45% Improvement on fairness!

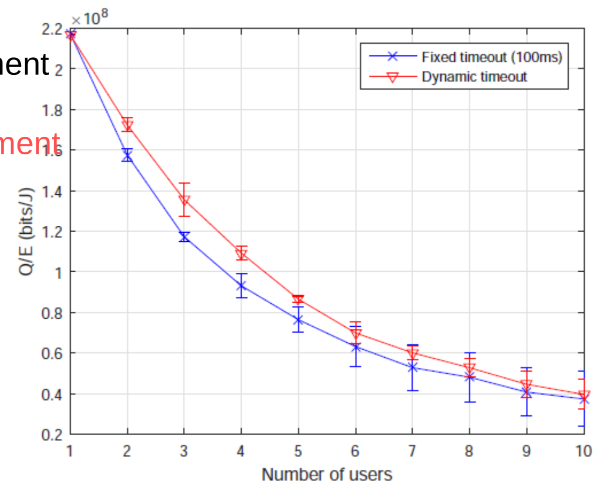


Fig. 10: Average Efficiency for DASH

Conclusion

- Cross-layer (APP, MAC) algorithm for Wi-Fi DPSM with DASH
- Use RTT and DPSM timeout to determine the next video quality and adapt timeout dynamically
- Low complexity, fully distributed
- For 10 users, improve power consumption by 15% and fairness by 45%, in the meanwhile, retain the video quality with better efficiency
- Future work: theoretical analysis and formulation; optimize the parameters; testbed; buffering strategy, etc.

Acknowledgment

- European Celtic-Plus project CONVINcE
- EC FP7 Marie Curie IAPP Project, MeshWise

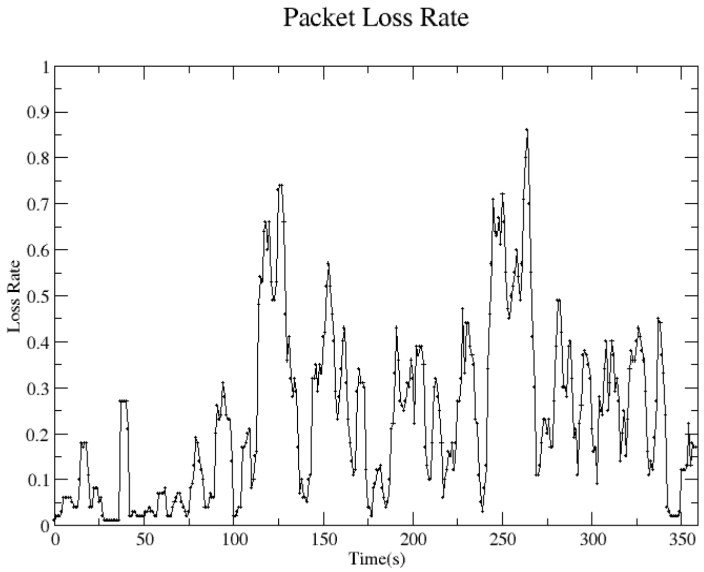
Outline

- –WiFi Packet Loss
- –Fountain Coding
- –Fountain Coding Transmission Protocol
- –Single Stream Power Consumption
- –Multiple Stream Power Consumption
- –Conclusion

Introduction 2

- Motivation
  - Wireless Networks (WiFi) have very lossy channels
  - High packets losses with retransmissions leads to increased power and energy consumption.
  - Fountain coding can reduce complexity and number of retransmissions.
- Contribution
  - Reliable UDP protocol for video streaming.
  - WiFi Packet Loss rate
  - Reliable video streaming protocol using fountain coding.

WiFi Packet Loss

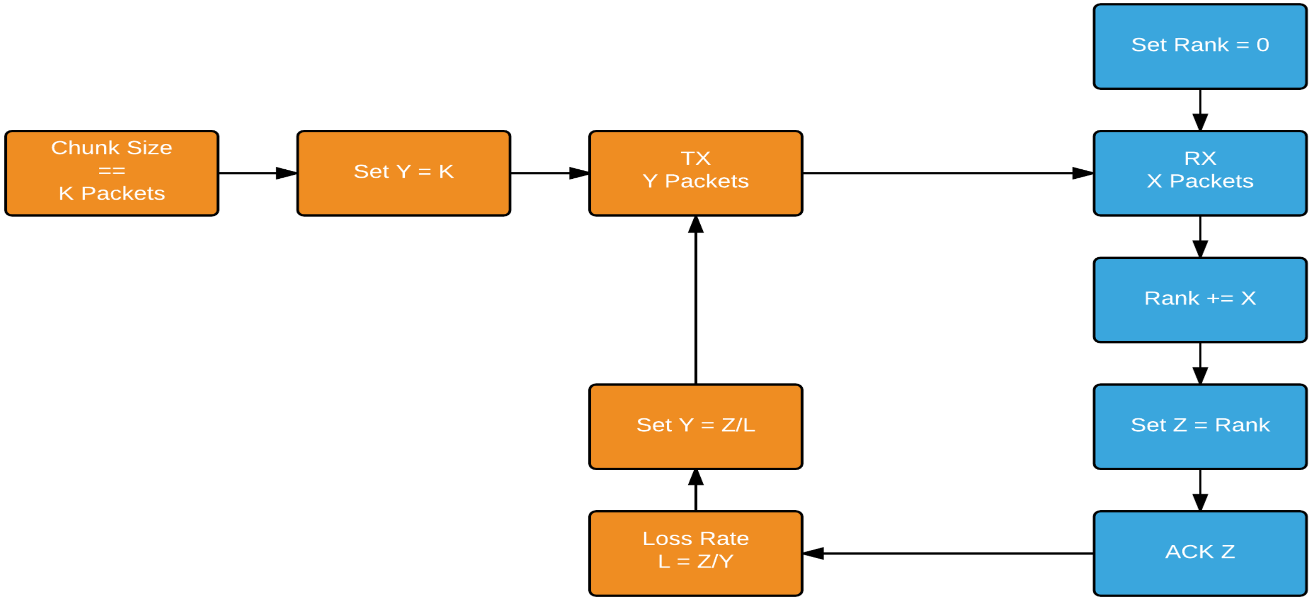


- WiFi typically experiences high packet losses.
- Not well known due to efficient MAC ARQ and rate adaptation.
- 5 min sample shows peaks of up 85% loss.
- Broadcast and Multicast packets are not retransmitted by MAC layer.

Fountain Coding

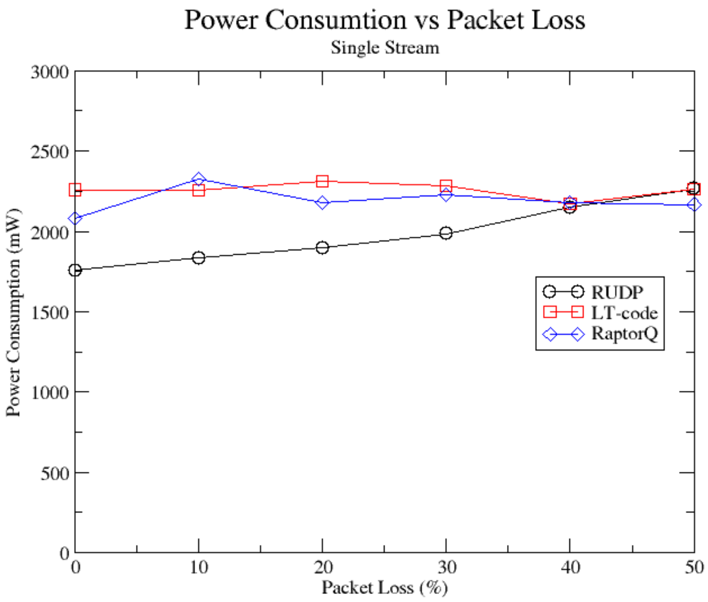
- Transmitted packets are linear combination of a set (chunk) of other packets.
- If set consists of *K* packets, ***any*** *K* packets can be received, regardless of order. No sequence numbering needed.
- LT (Luby Transform) combines *d* packets, chosen from an Ideal Soliton Distribution. ~1 % risk of linear dependent packets.
- Raptor Q codes wraps the LT code with an outer LDPC code. ~0.01% of linear dependent packets.

Fountain Coding transmission protocol



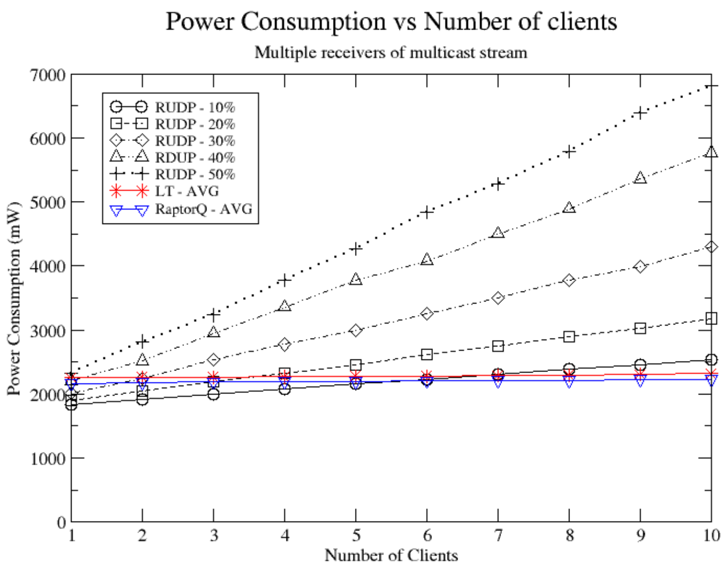
- Transmit K encoded packets based on chunk size.
- ARQ based on current rank, Z at receiver (in ACK).
- Estimate loss rate, L based on Z and number tx packets.
- Transmit extra packets based on L.

FT coding single stream



- For low packet loss rates the encoding and decoding computation overhead is higher than the reduced number of retransmissions.
- For very packets loss rate a slight gain may be achieved.
- For UDP based streaming, power consumption increases with packet loss rate.
- For FT coding, power consumption stable.

FT coding up to 10 streams



- WiFi typically experiences high packet losses.
- Not well known due to efficient MAC ARQ and rate adaptation.
- 5 min sample shows peaks of up 85% loss.
- Broadcast and Multicast packets are not retransmitted by MAC layer.

Conclusion

- R-UDP based protocol for reliable video streaming.
- Fountain Coding extensions to the R-UDP protocol.
- Fountain Coding for single streams may improve power consumption for very high loss rates, but overhead increases consumption for low loss rates.
- Fountain Coding for multiple streams improves power consumption between 14% and 205%.

Thank you!

Questions?

anders@telhoc.se

zhi.zhang@eit.lth.se