

WiFi Mobility without Fast Handover

Costin Raiciu
Jointly with Andrei Croitoru and Dragoș Niculescu

University Politehnica of Bucharest

WiFi mobility is back in fashion

Cellular data growing at a rate that is not sustainable in the long run

Offloading to WiFi touted as a solution

Ubiquitous Access Point deployments in urban areas

AP deployment in Bucharest Center



L2 Mobility mantra

One AP at a time

- When link fades, look for new Aps
- Scan for new APs
- Connect to one of them based on
  - Signal strength
  - **User preference**
  - Estimated capacity

FAST HANDOVER
lots of research

Client doesn't know best AP until it tries it

Best AP changes when client is mobile or static

Why bother selecting best AP?

Connect to all APs

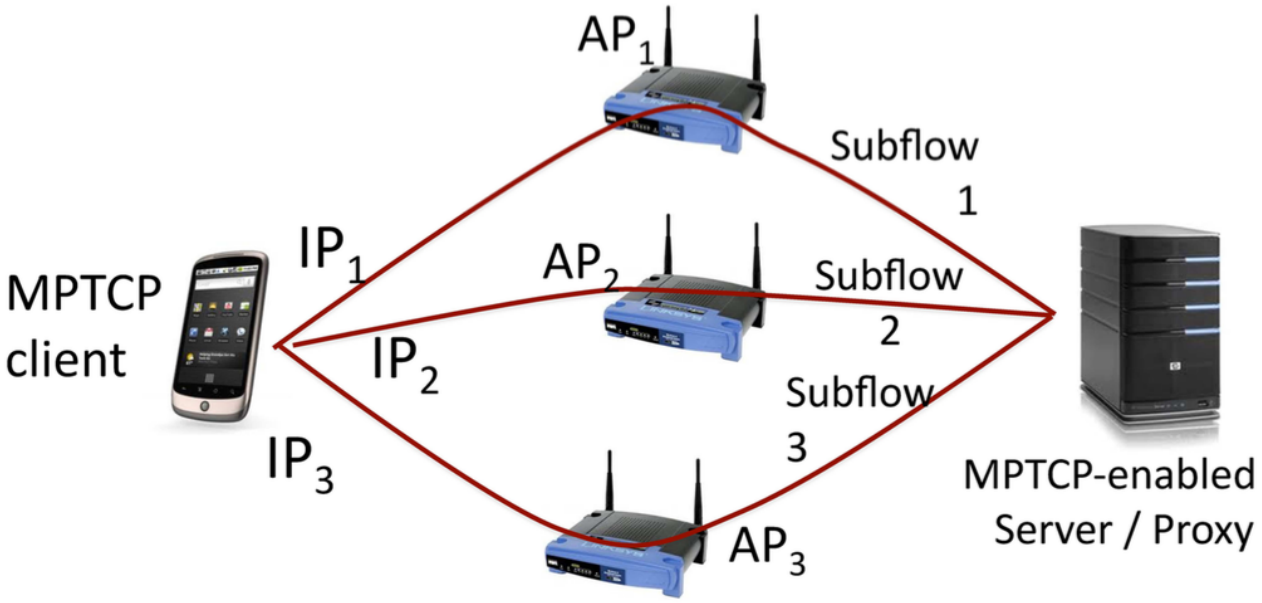
Spread traffic with Multipath TCP

while having a single NIC in the client

Implementation is straightforward

- On a single channel, use virtual interfaces
- Channel switch between different channels

Connect to all APs



So how should traffic be allocated?

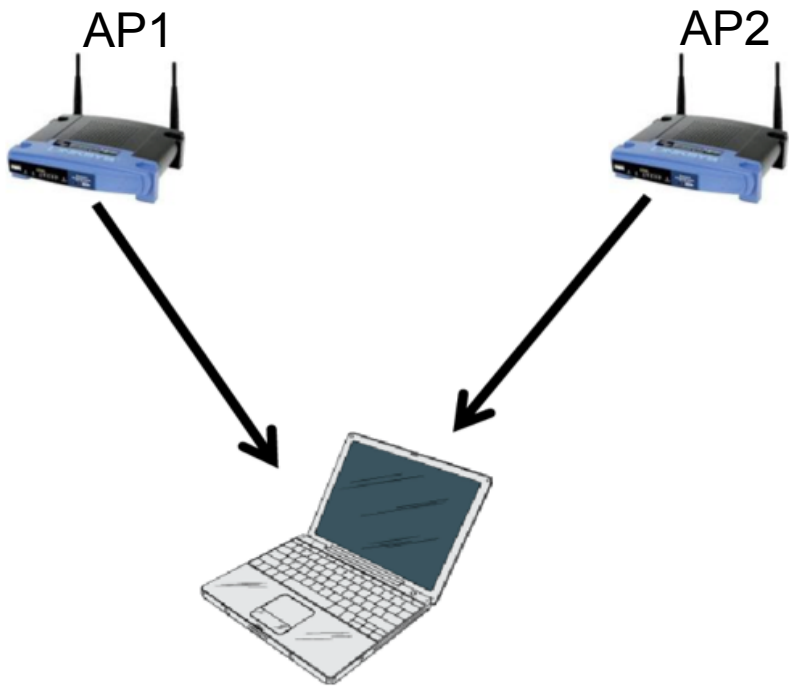
Most traffic should come via the best AP while probing all other APs

Multipath TCP + Multiple APs

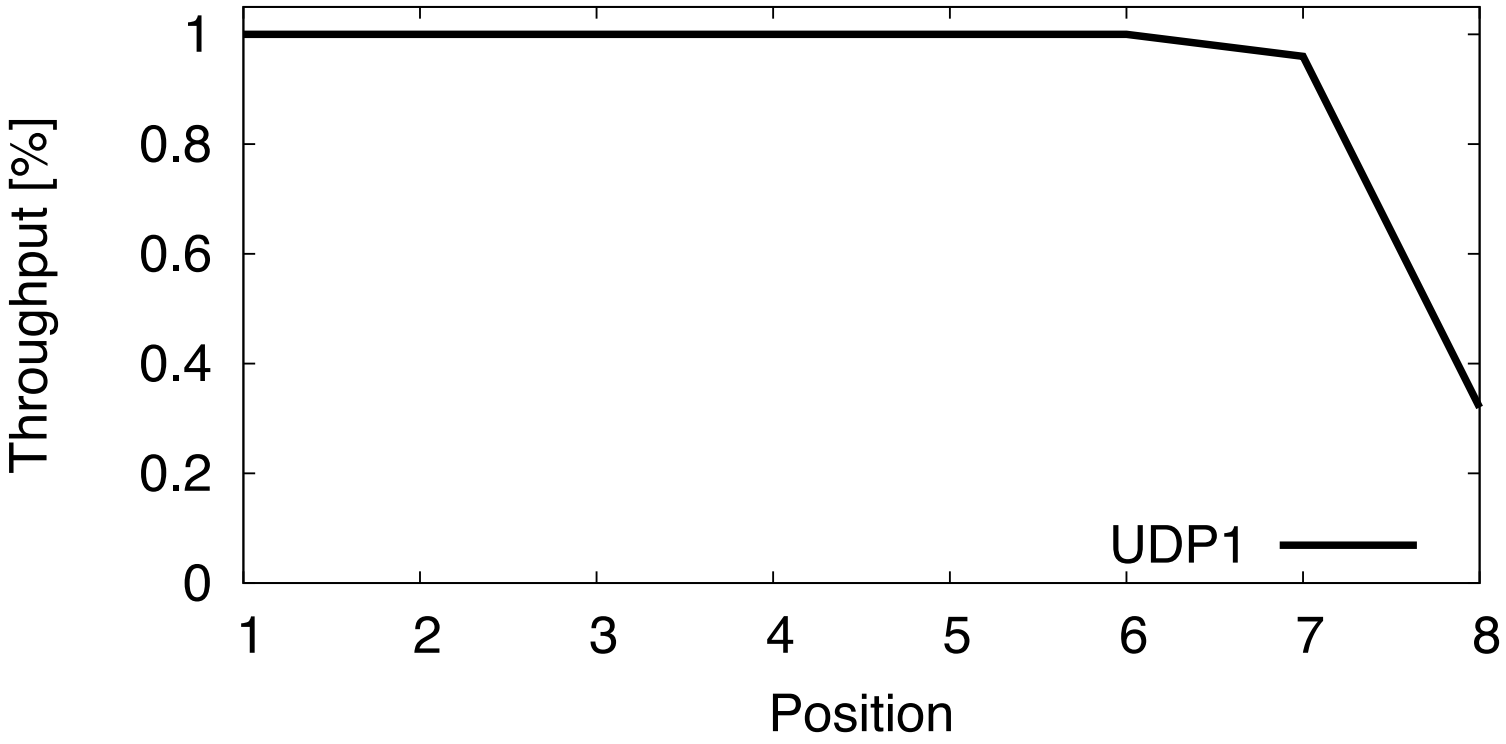
Experiments

- Single channel
  - Hidden terminal
  - Carrier-sense
- Multiple channels
  - Use channel switching

    [see paper]



Hidden terminal Experiment

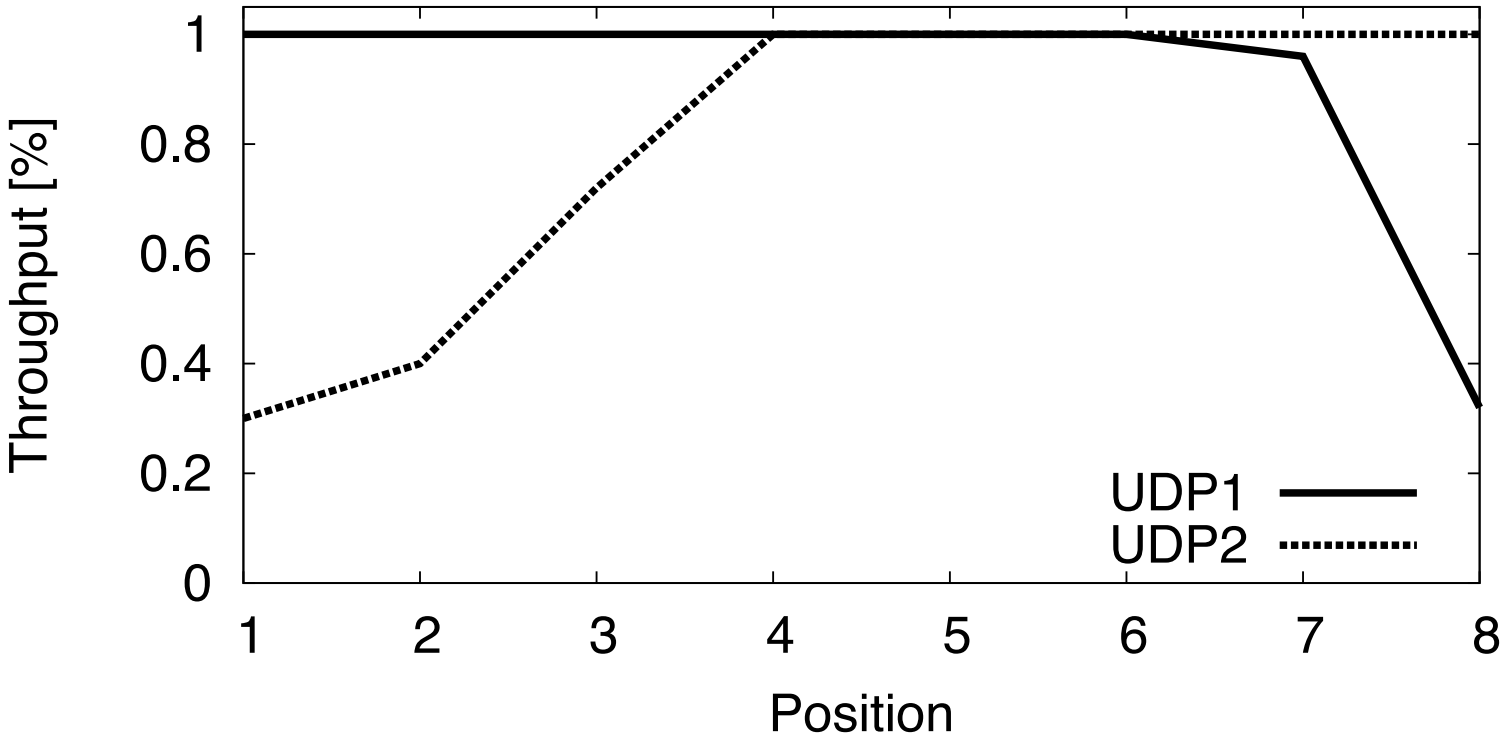


Client near AP1

In-between APs

Client near AP2

Hidden terminal Experiment

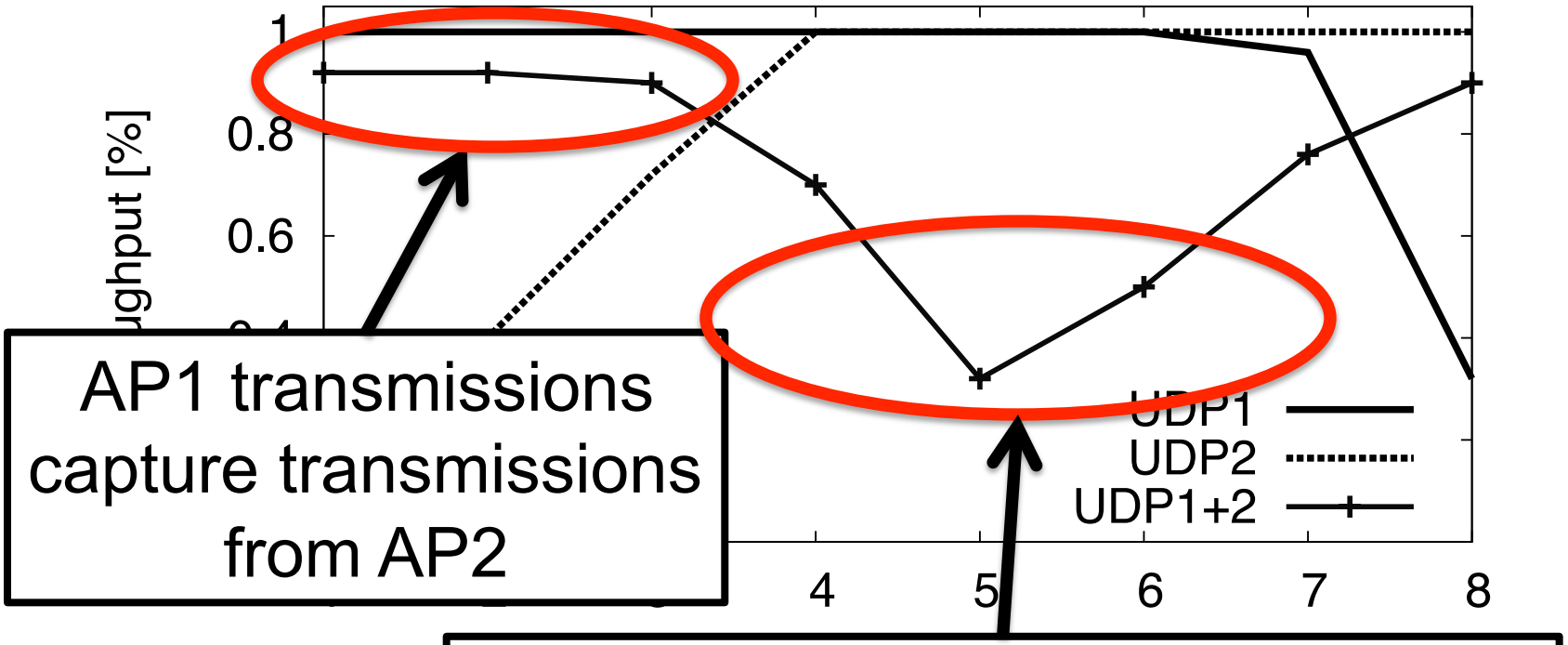


Client near AP1

In-between APs

Client near AP2

Hidden terminal Experiment



AP1 transmissions capture transmissions from AP2

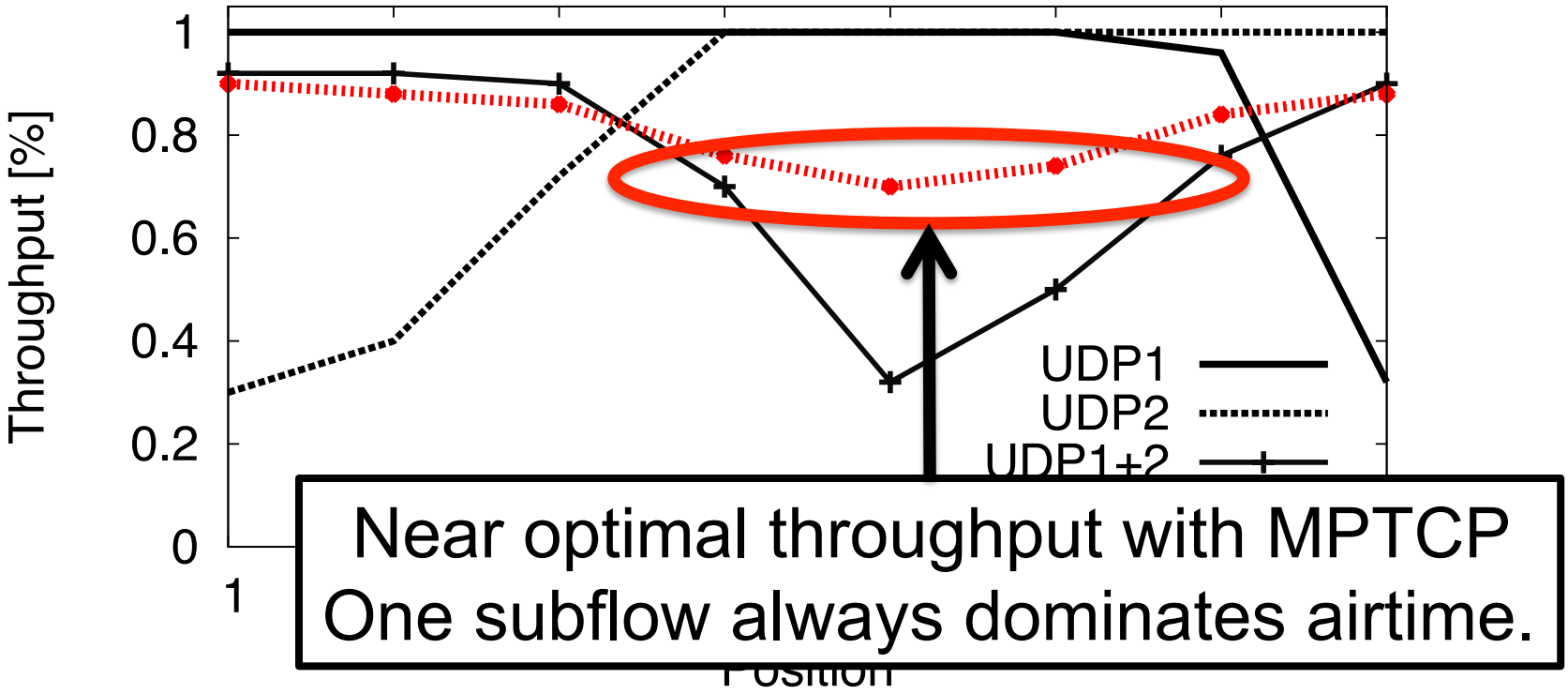
UDP throughput drops significantly

Client near AP1

In-between APs

Client near AP2

Hidden terminal Experiment



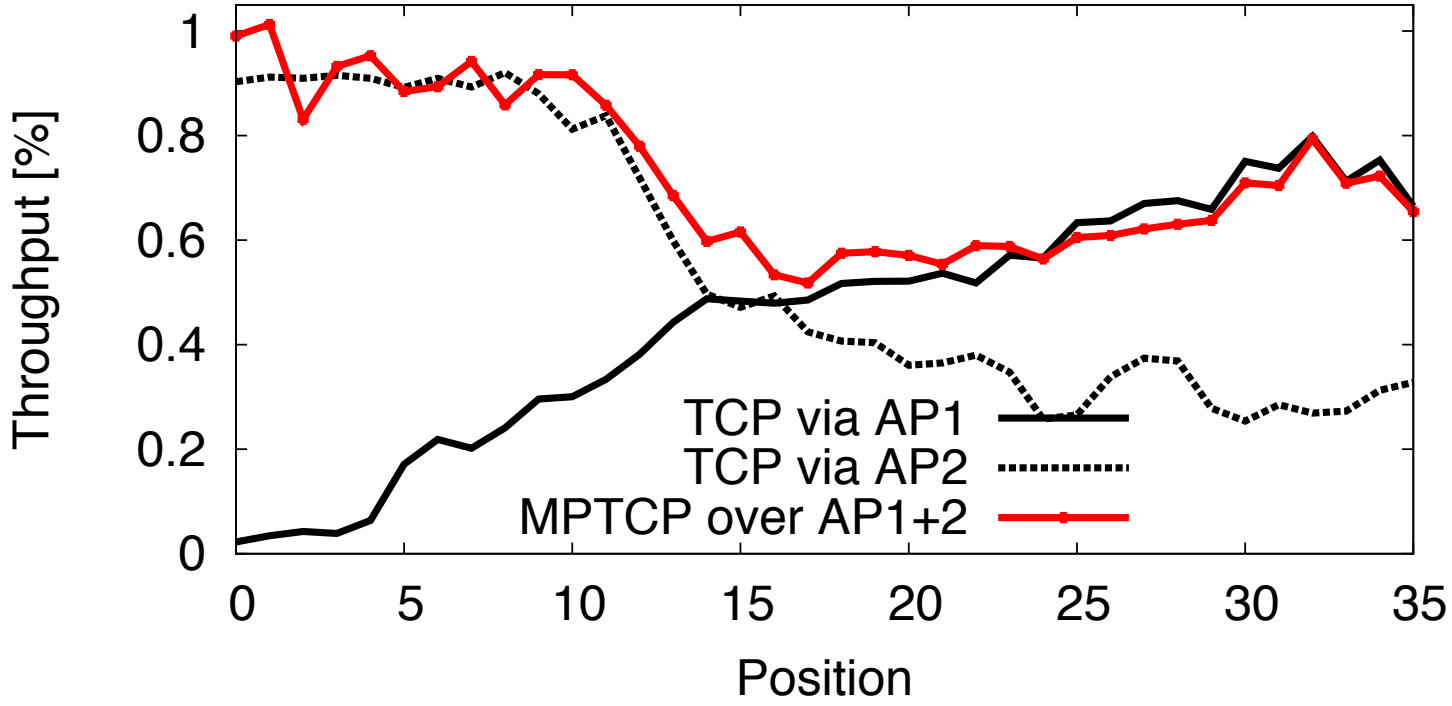
Near optimal throughput with MPTCP
One subflow always dominates airtime.

Client near AP1

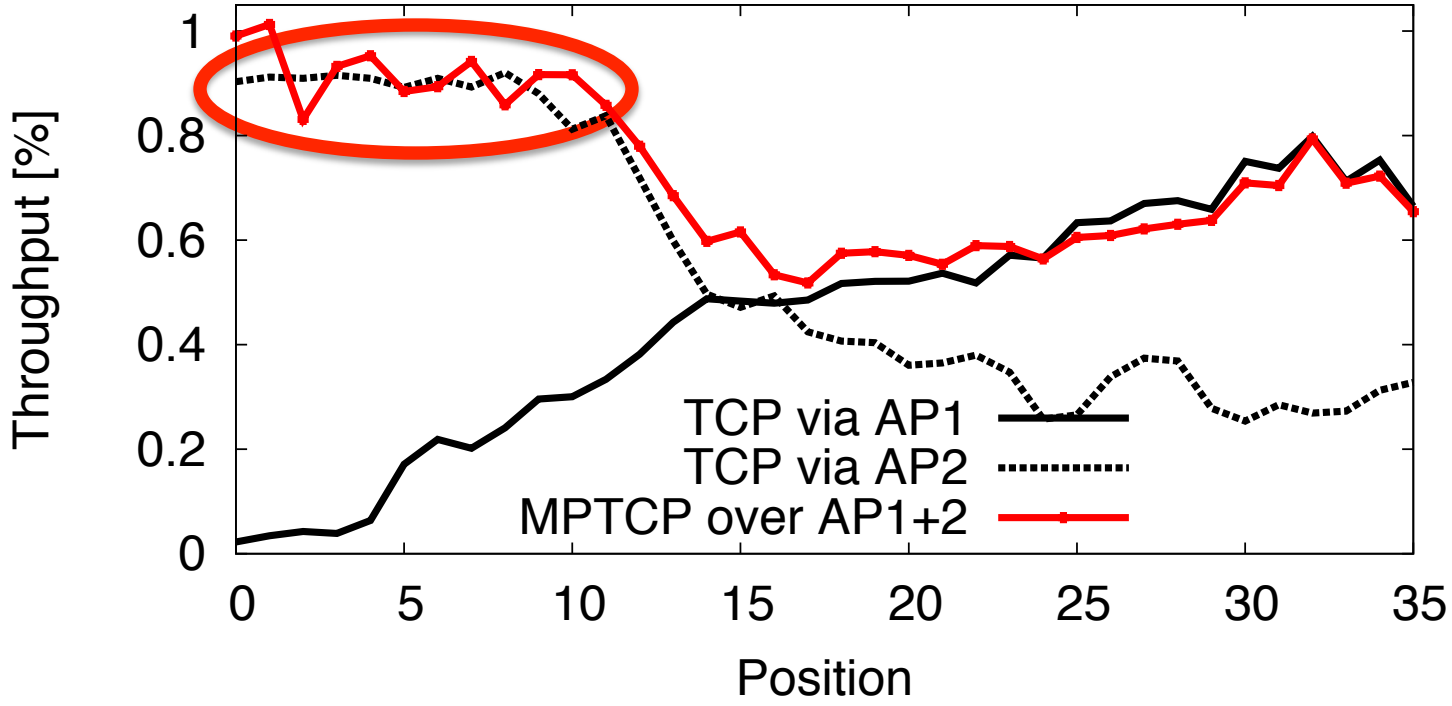
In-between APs

Client near AP2

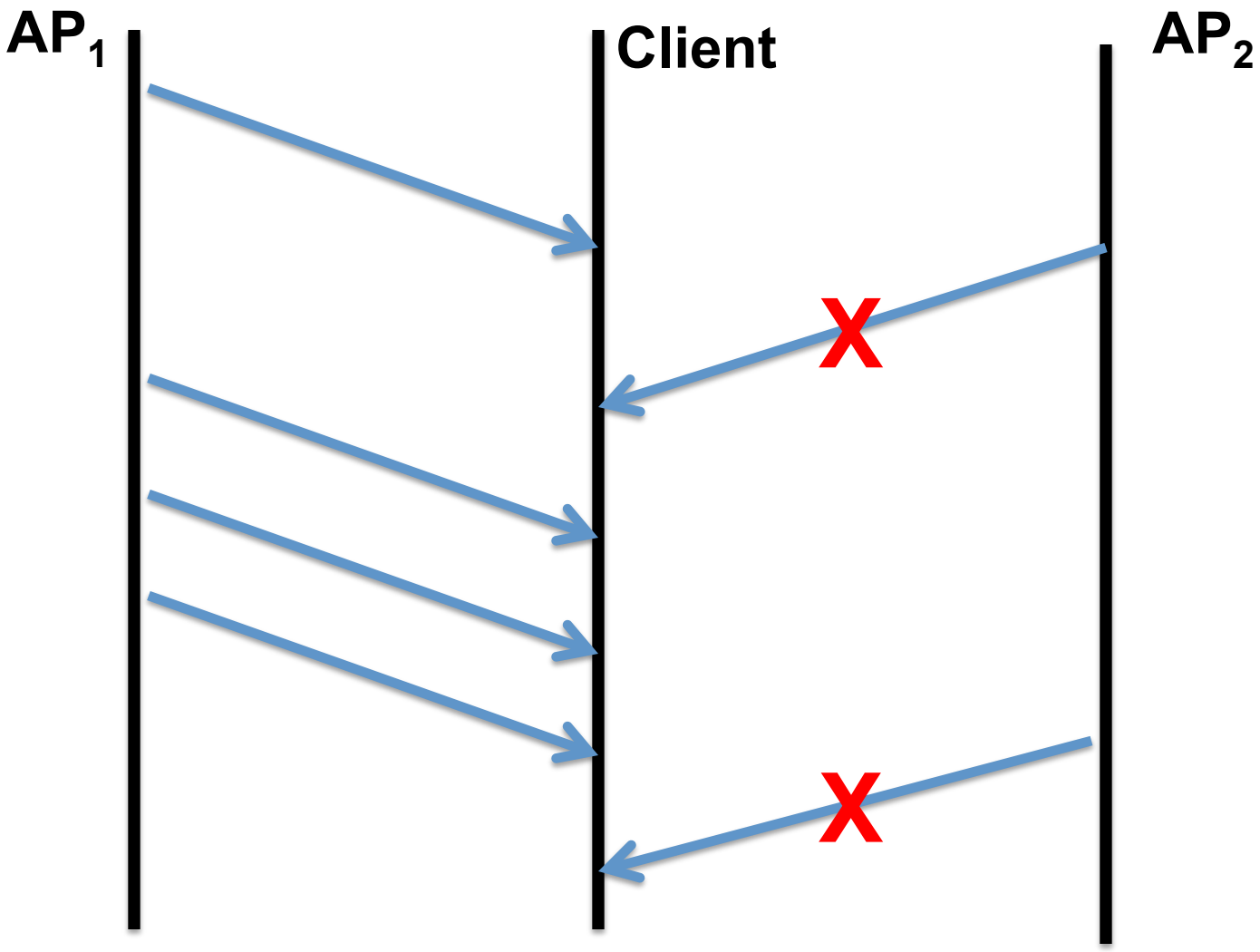
Carrier Sense experiment
APs hear each other



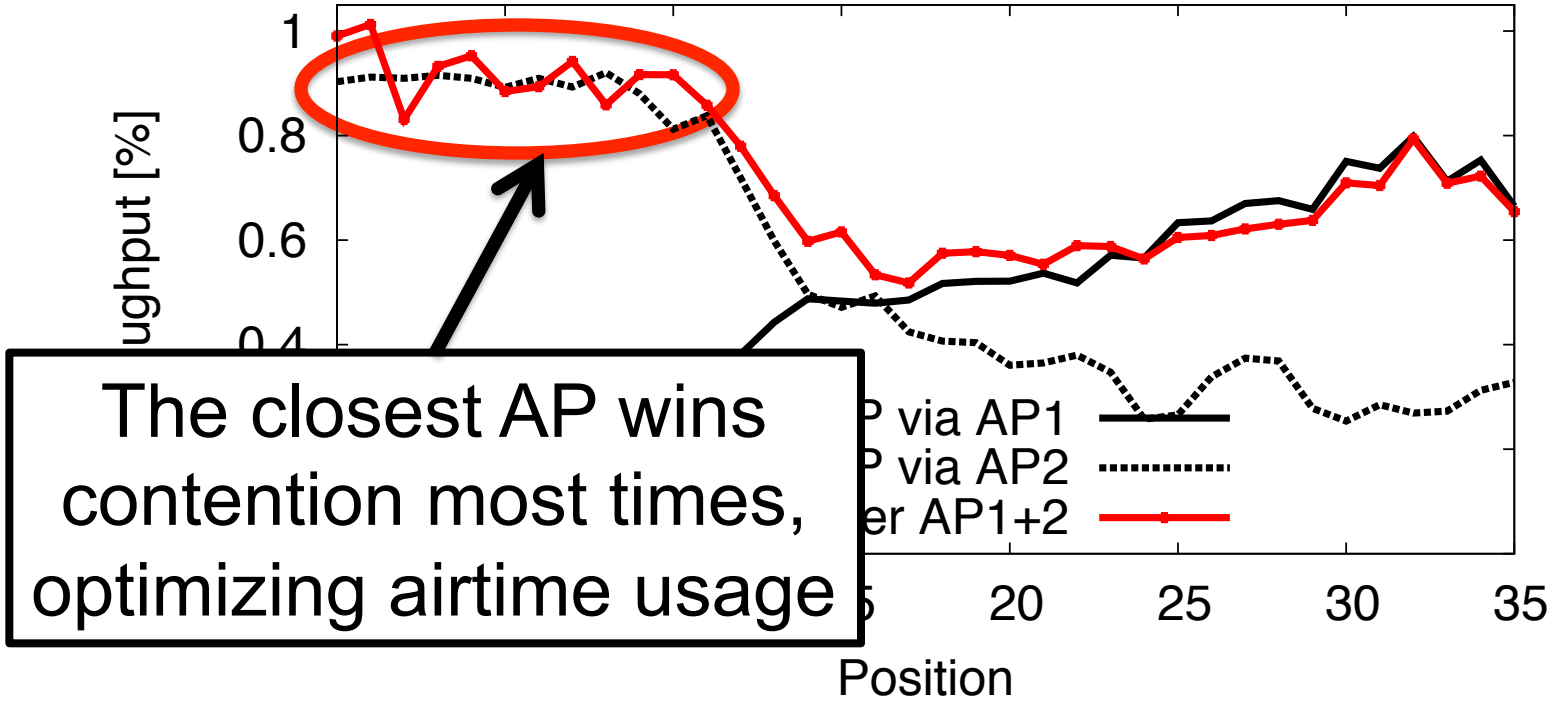
Carrier Sense experiment
APs hear each other



MAC with two APs sending

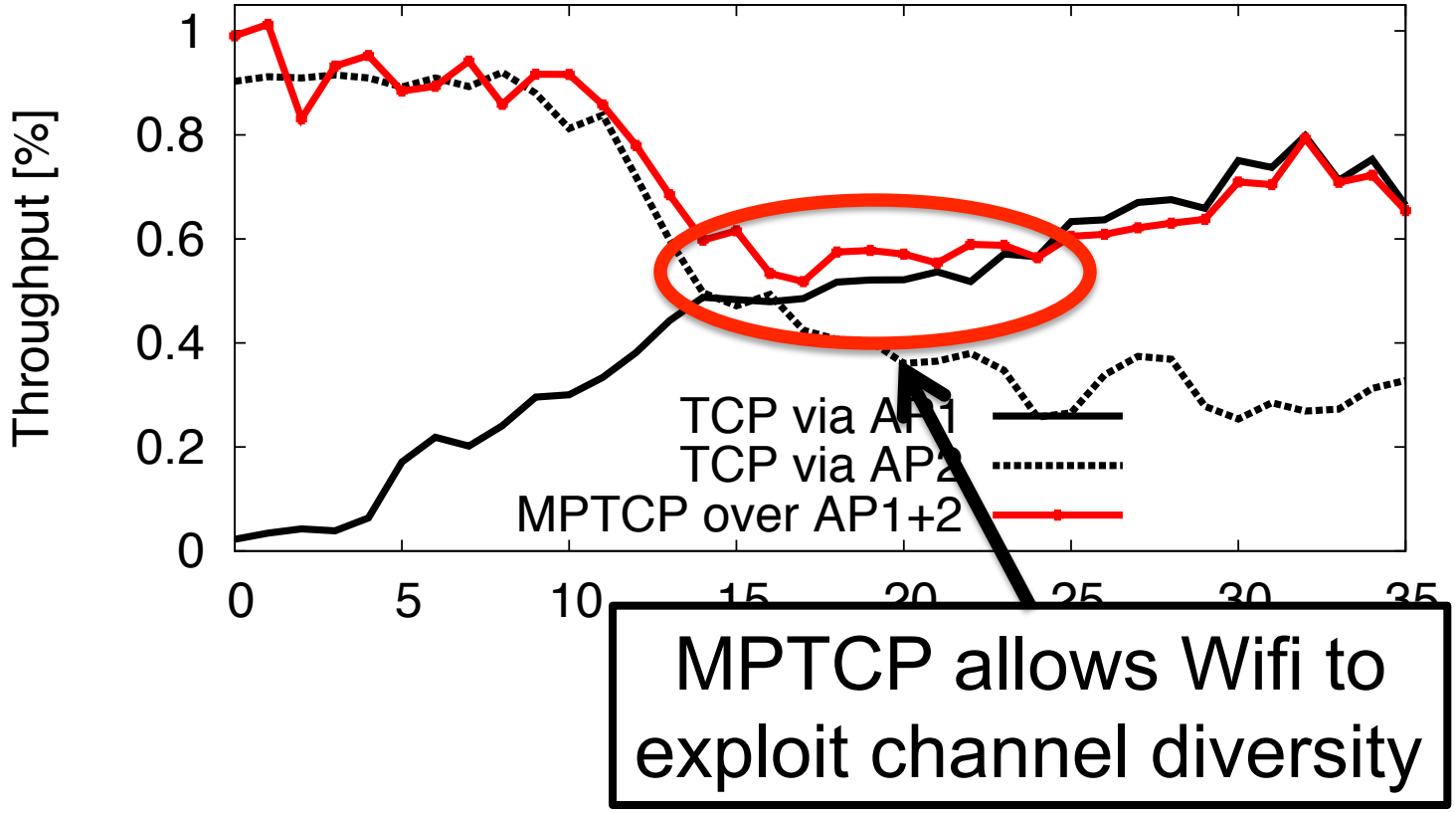


Carrier Sense experiment
APs hear each other



The closest AP wins contention most times, optimizing airtime usage

Carrier Sense experiment
APs hear each other

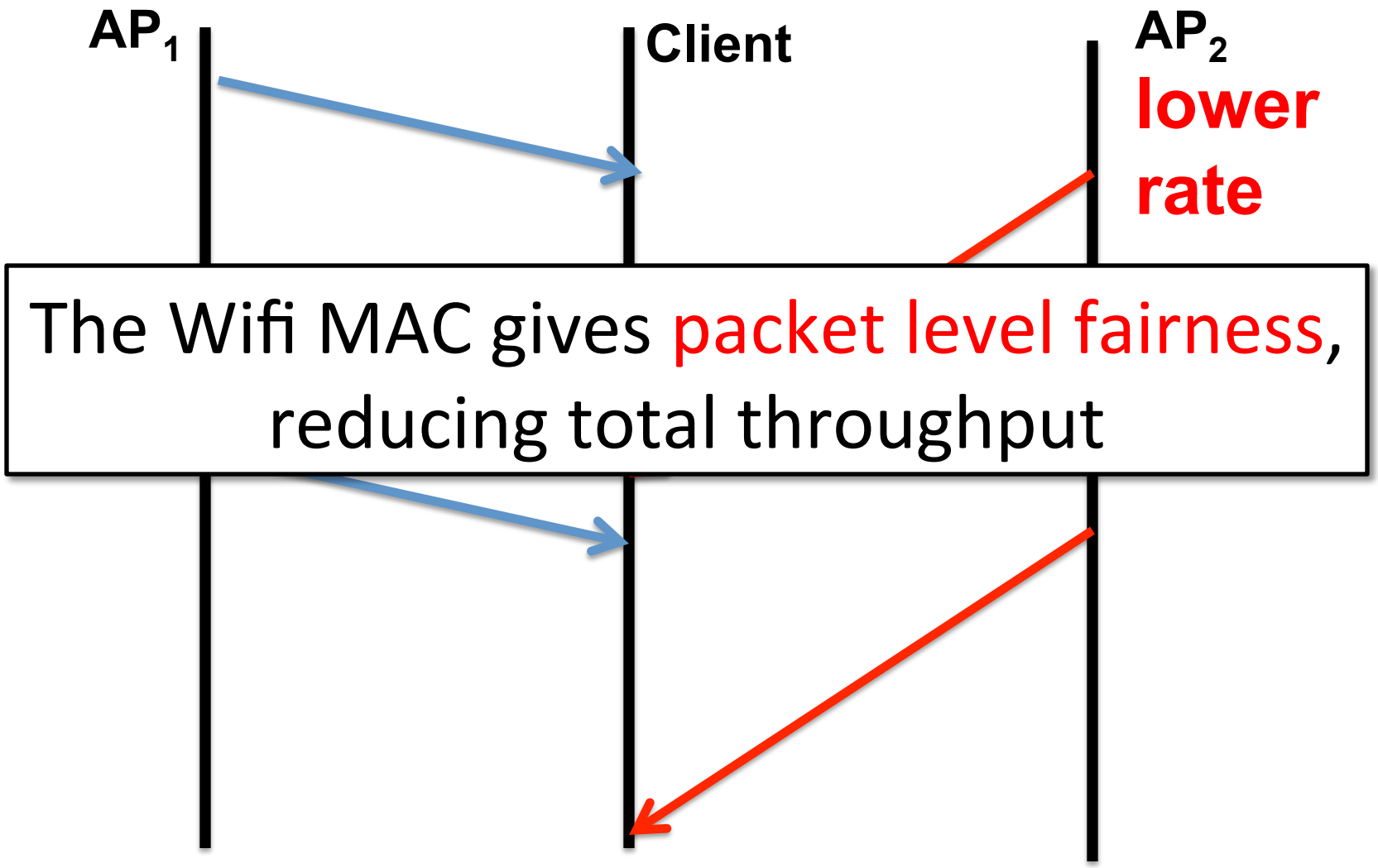


Great. Are we done?

Great. Are we done? **NO**

With 802.11n, MPTCP client gets half the throughput it should receive

MAC with two APs sending



Carrier-Sense performance depends on rate control algorithm

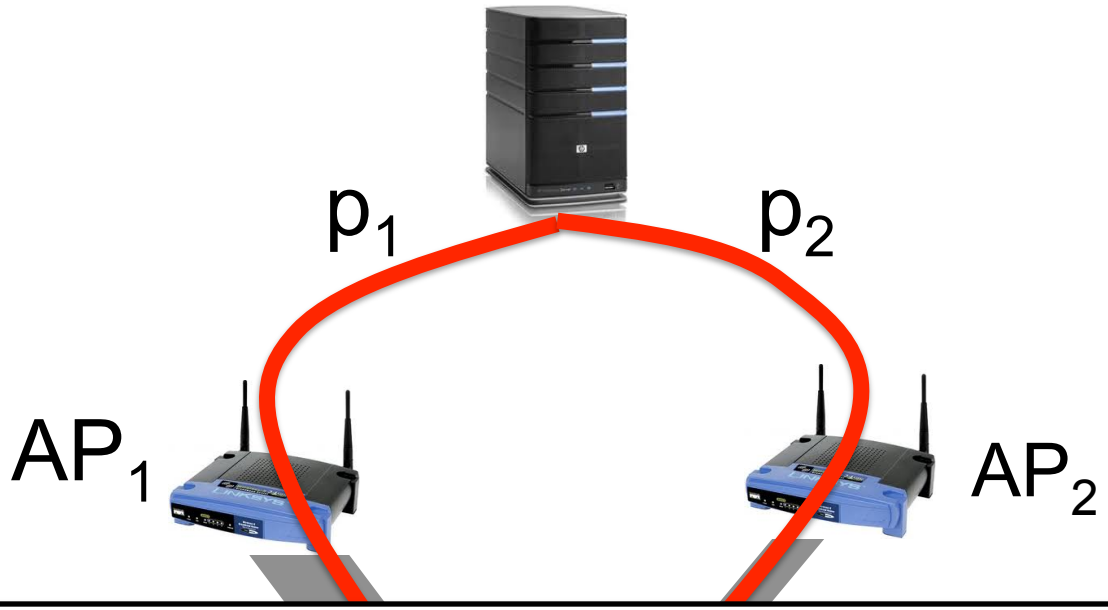
- Near-optimal behaviour when:
  - All APs used the same fixed rate
  - Rate control algorithms are specific to each AP vendor. Change is very difficult.
  - ux)
- Bad behaviour when:
  - Using less aggressive rate control algorithms
  - Using *minstrel-ht* (802.11n)

MPTCP congestion control primer

Move traffic away from congested links

- Put most data on the subflow with the smallest loss rate
- Subflows with higher loss rate only receive probe traffic

MPTCP congestion control in WiFi



When the MAC gives packet level fairness,

$p_1 = p_2$



Client knows which AP is better

Client estimates the average time T_i it takes AP_i to send one packet

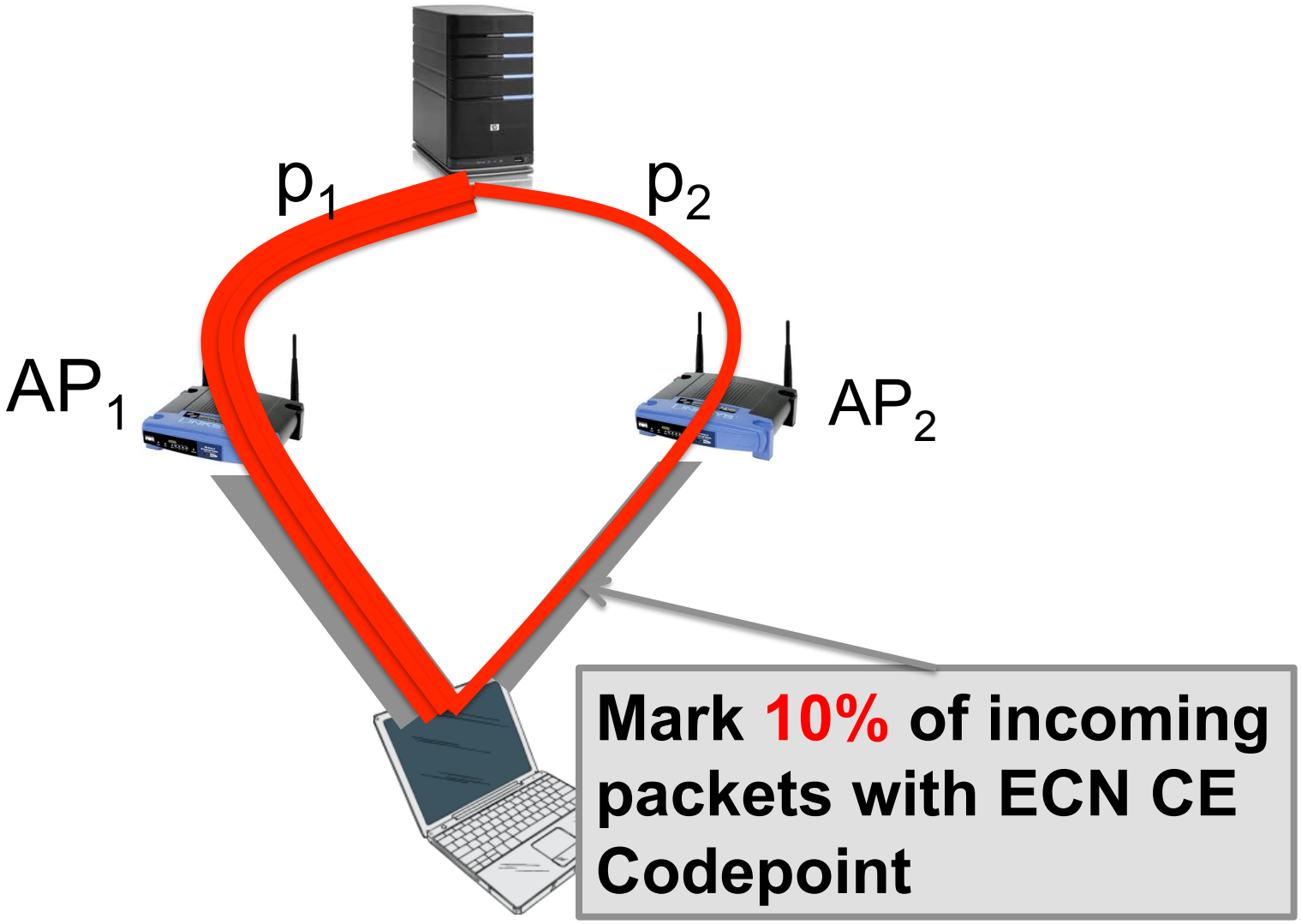
- Assuming AP is alone accessing the medium
- Passively estimate PHY loss rates [see paper]

Client orders its APs according to T_i

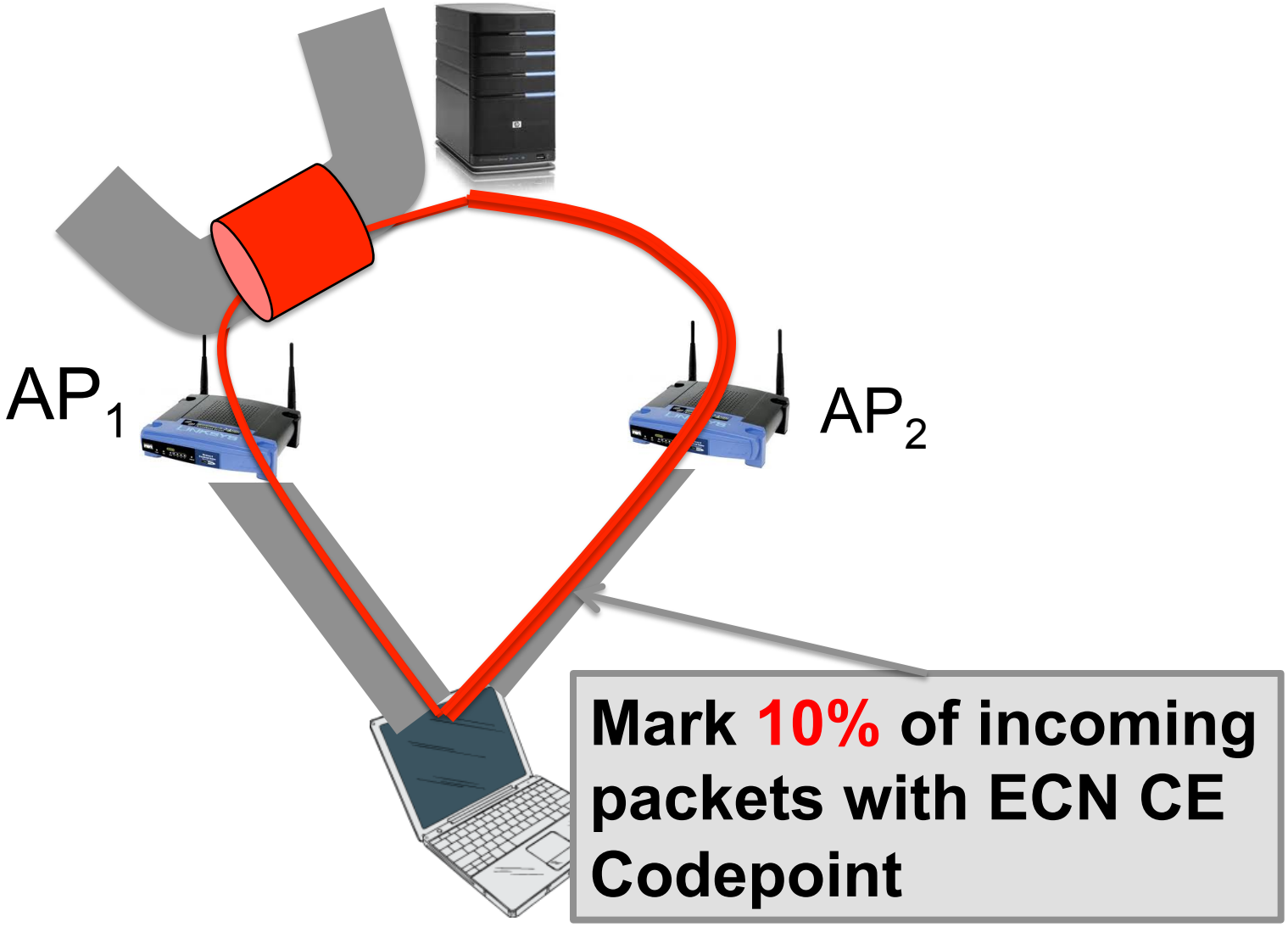
- Informs server that AP_j is bad when

$$T_j > 1.3 \min_{i=1,N}(T_i)$$

Client ECN marking to steer traffic



Client ECN marking to steer traffic



MPTCP congestion control primer

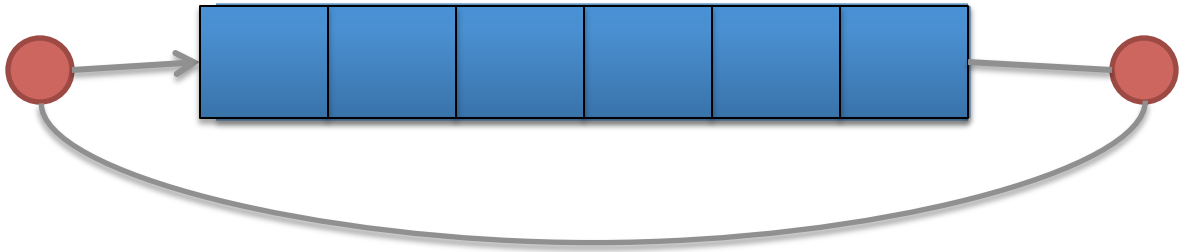
Move traffic away from congested links

- Put most data on the subflow with the smallest loss rate
- Subflows with higher loss rate only receive probe traffic

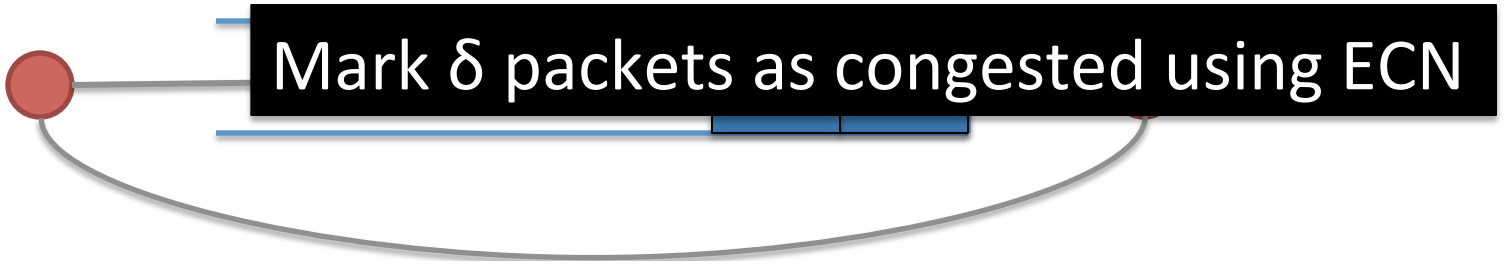
Do at least as good as TCP on the best path

- Use RTT and loss estimates on each subflow to estimate what TCP would get

Safe ECN Marking



$$B = \sqrt{\frac{2}{p}} \cdot \frac{MSS}{RTT}$$



$$B = \sqrt{\frac{2}{p}} \cdot \frac{MSS}{RTT} = \sqrt{\frac{2}{p+\delta}} \cdot \frac{MSS}{RTT_\delta}$$

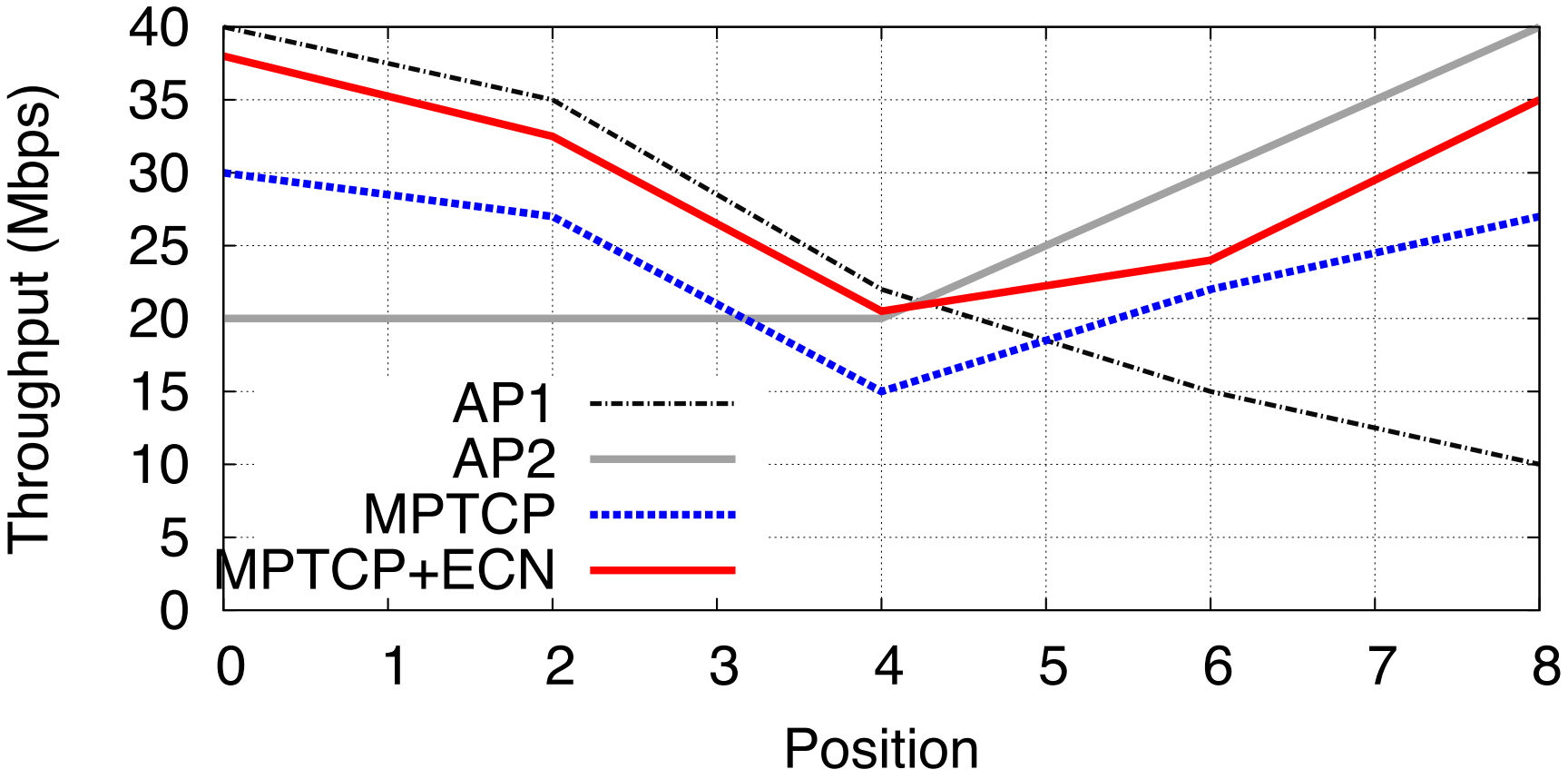
Safe ECN Marking

The client can compute the safe marking threshold delta:

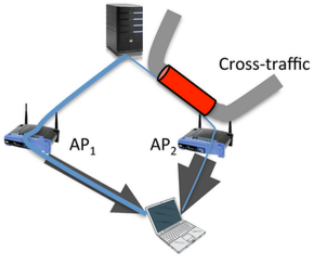
$$\delta = \frac{1}{2} \cdot \left(\frac{50 \cdot T_1^2}{RTT \cdot (RTT - 50 \cdot T_1)}\right)^2$$

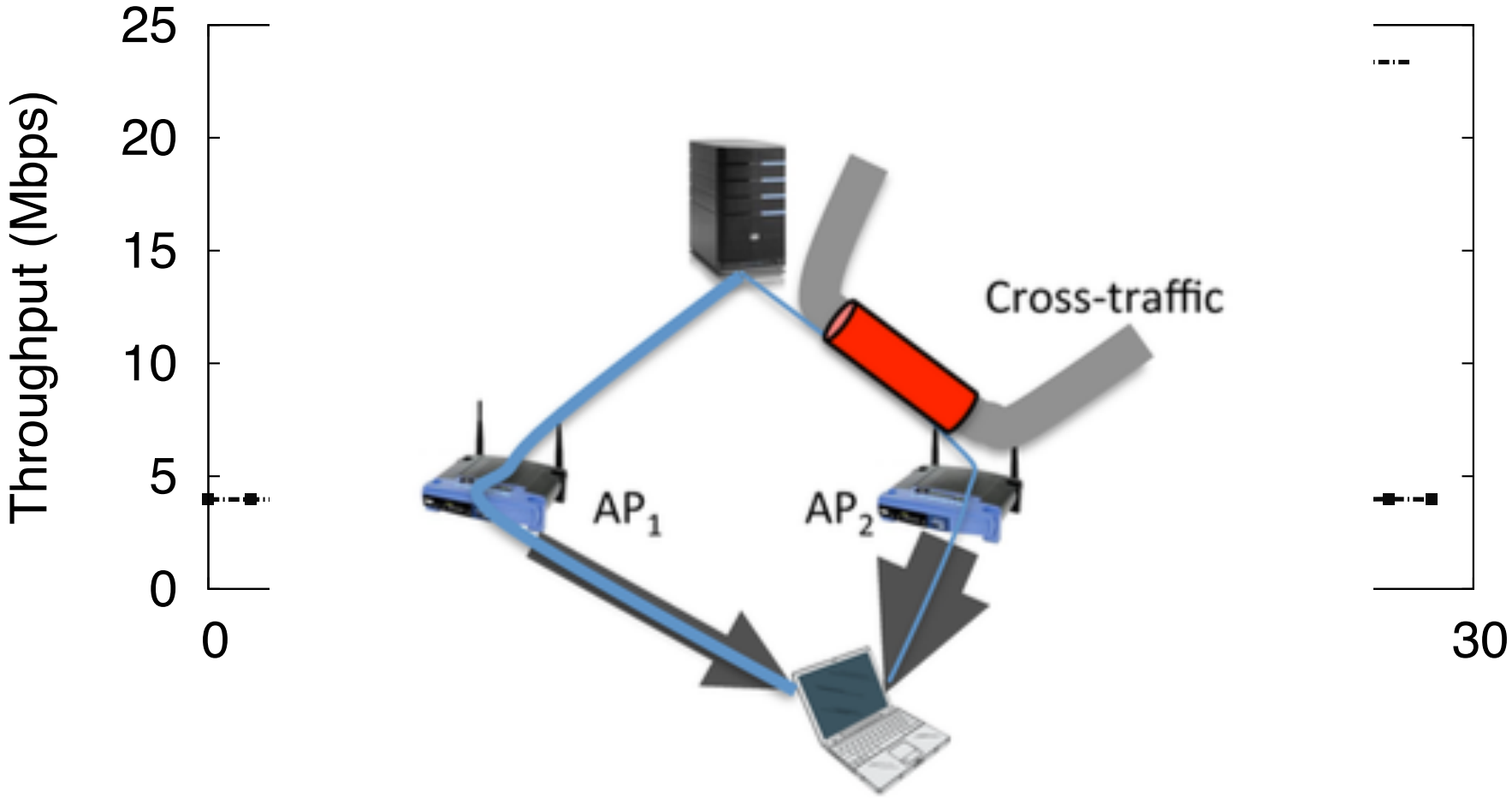
Implemented in device-independent part of the WiFi driver of the Linux kernel.

How well does ECN marking work for 802.11n?

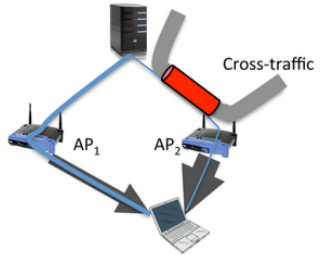


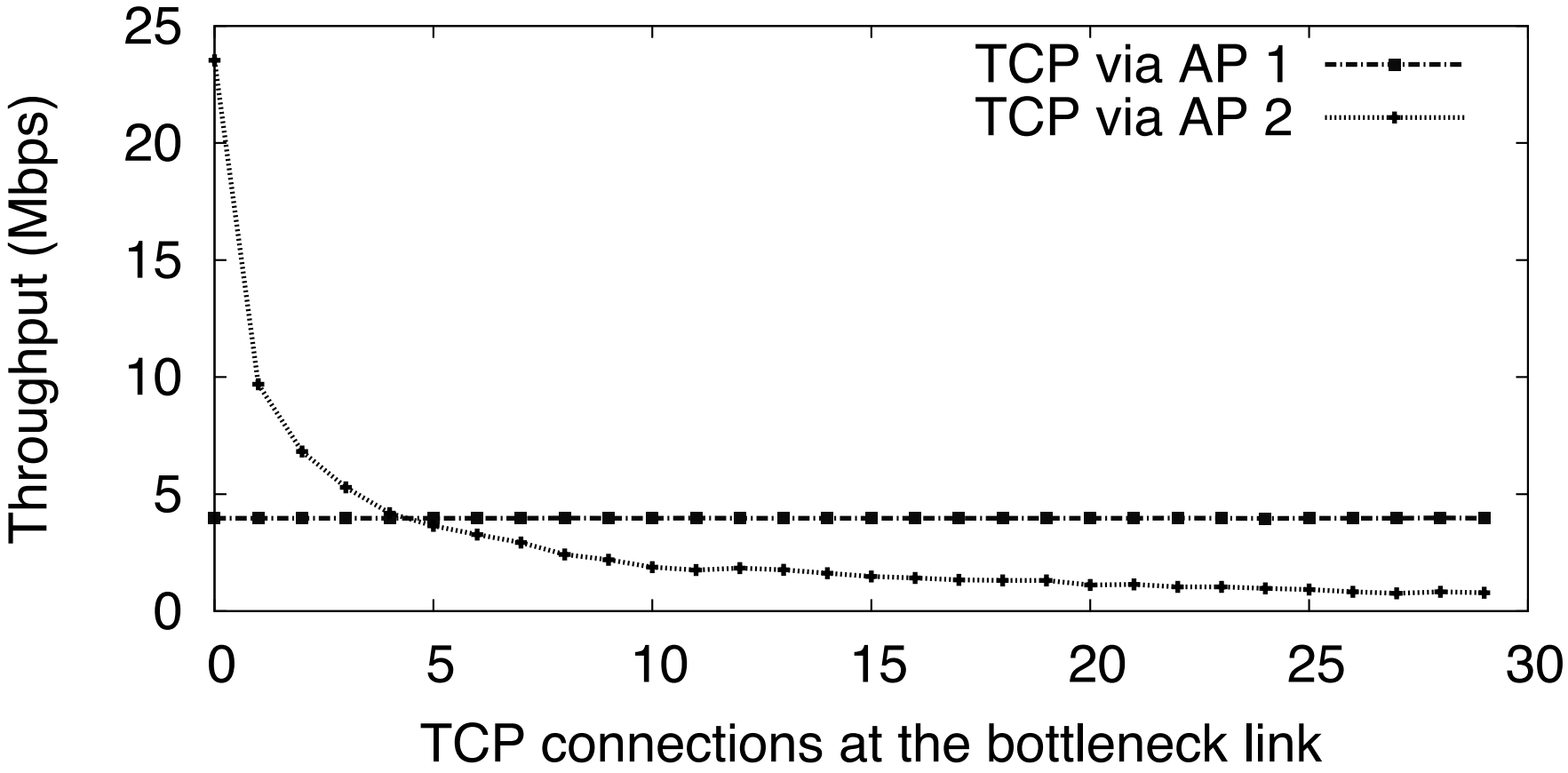
Ensuring good performance with bottlenecked APs



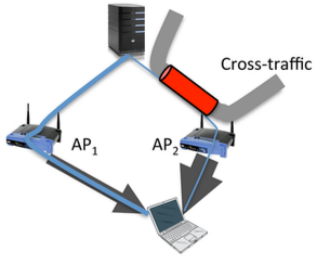


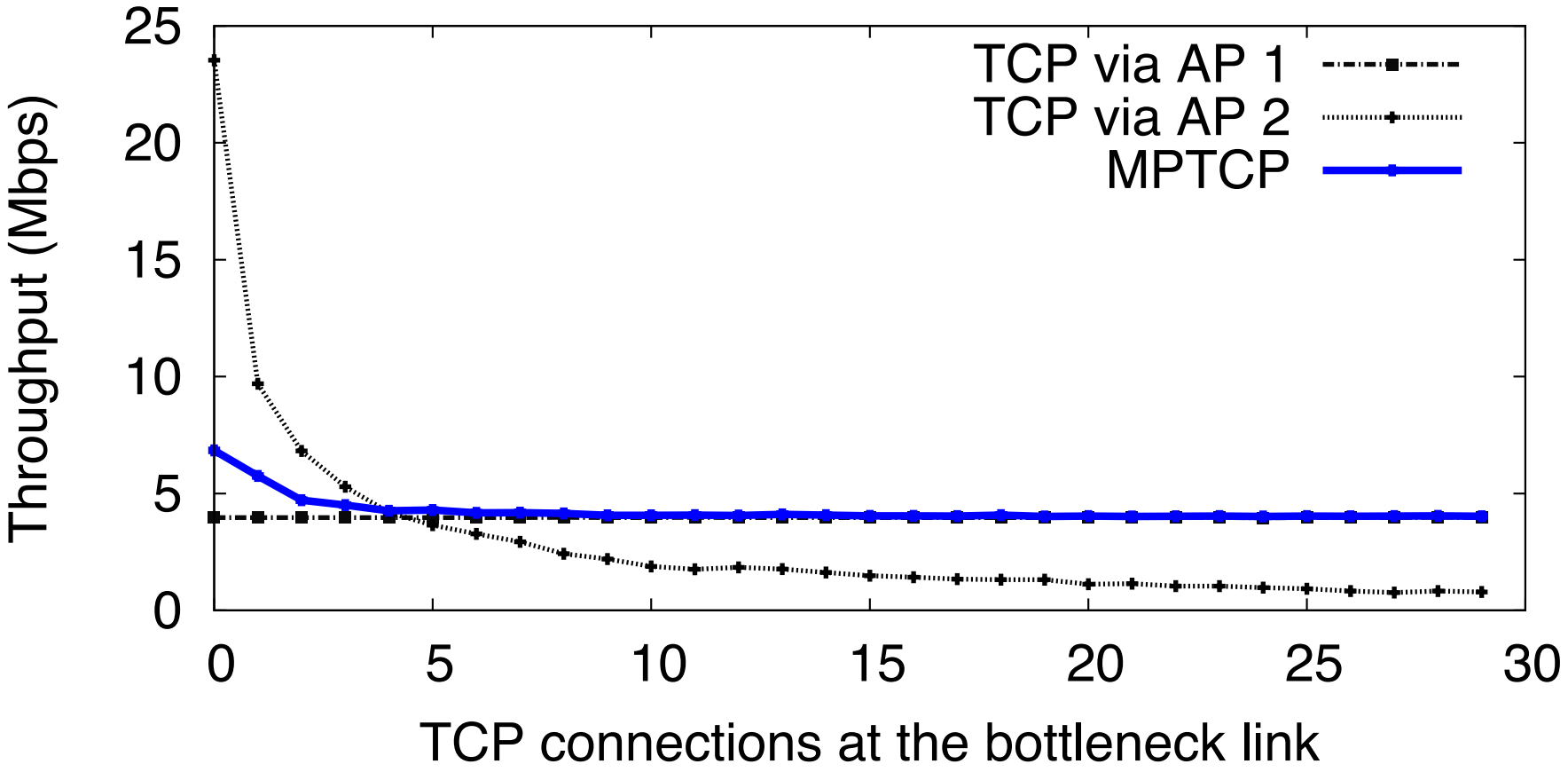
Ensuring good performance with bottlenecked APs



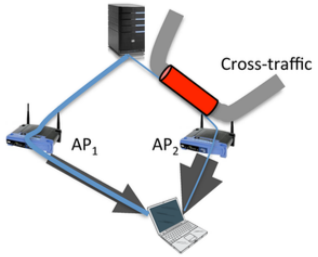


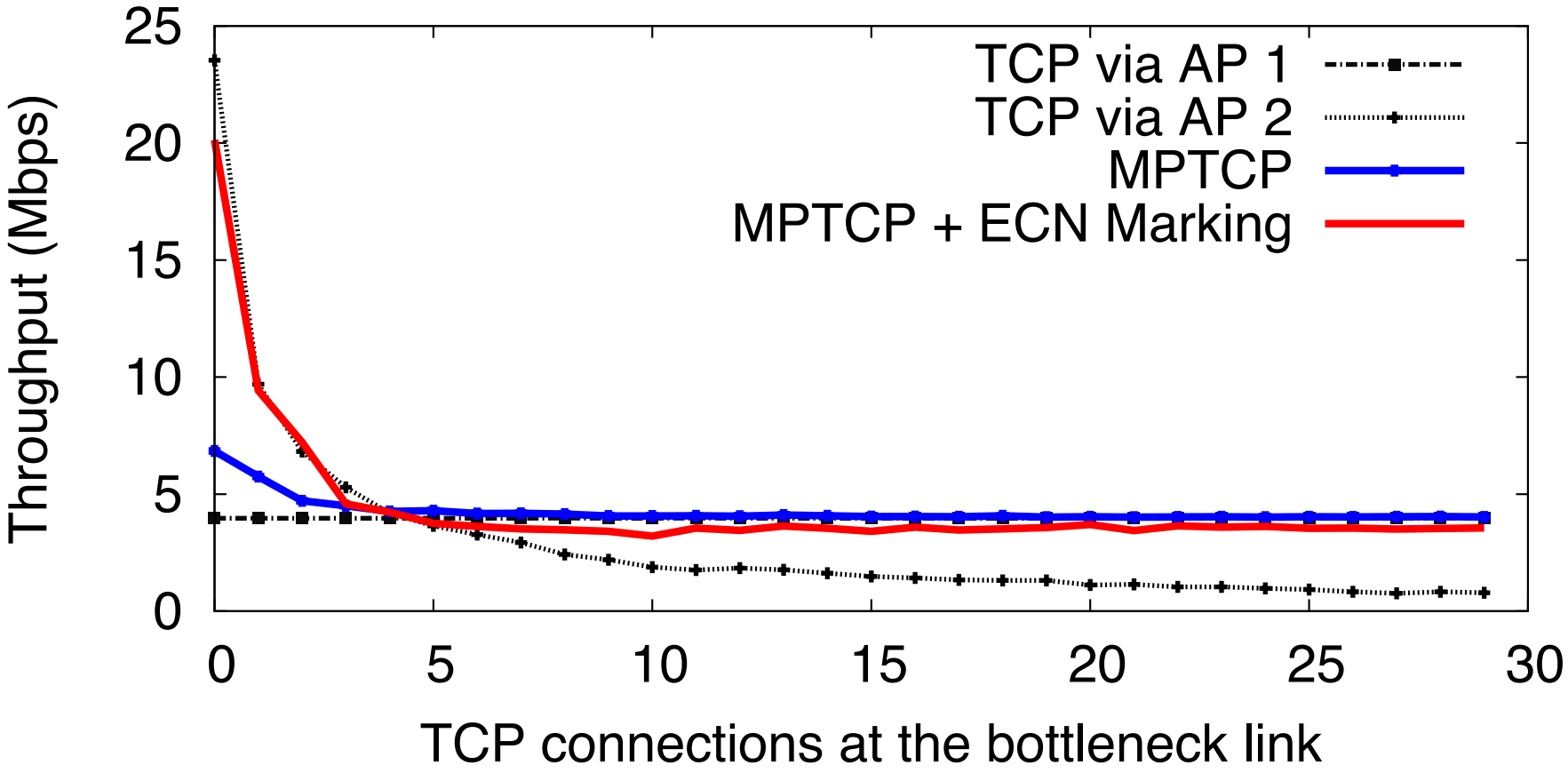
Ensuring good performance with bottlenecked APs



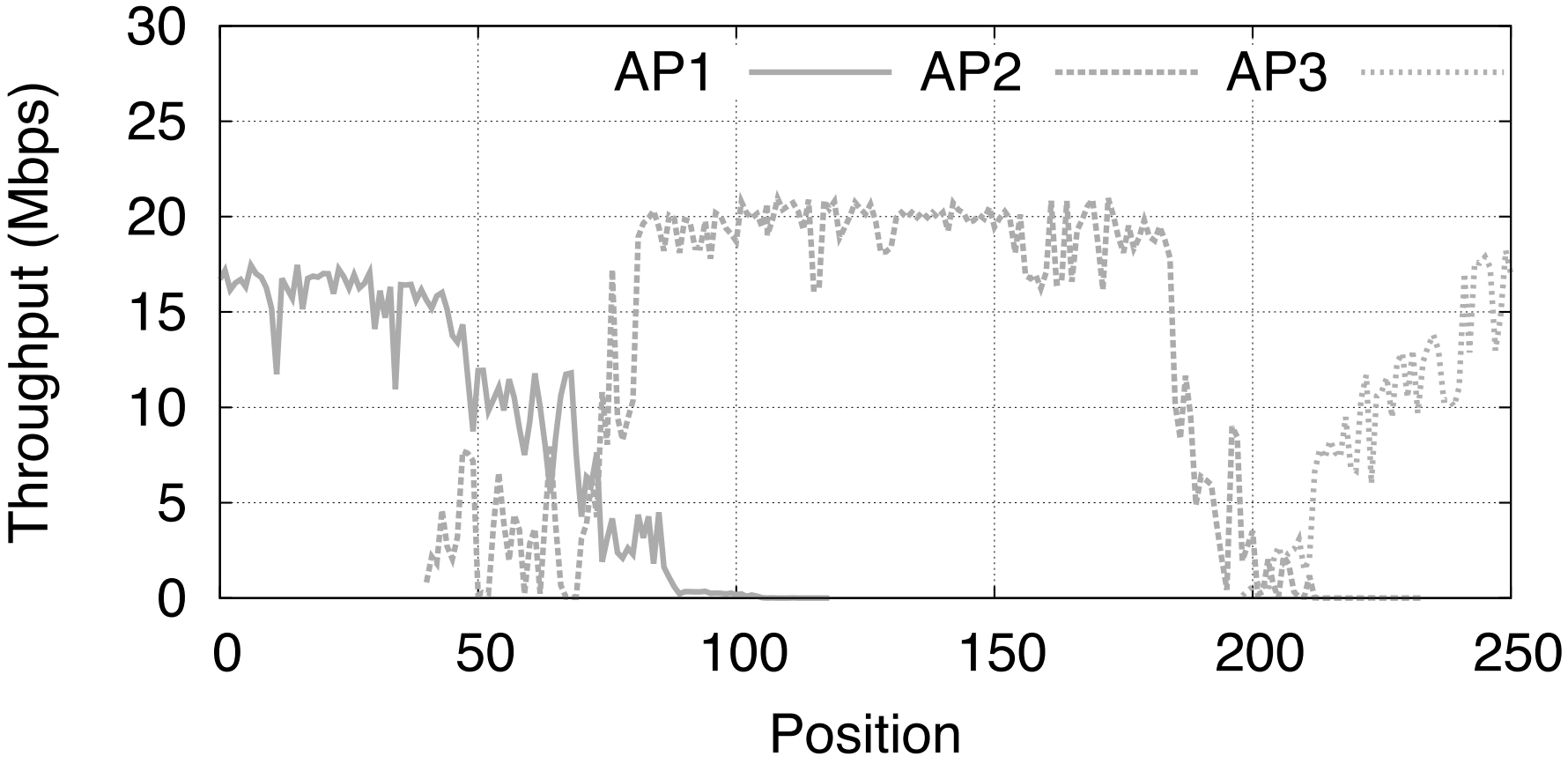


Ensuring good performance with bottlenecked APs

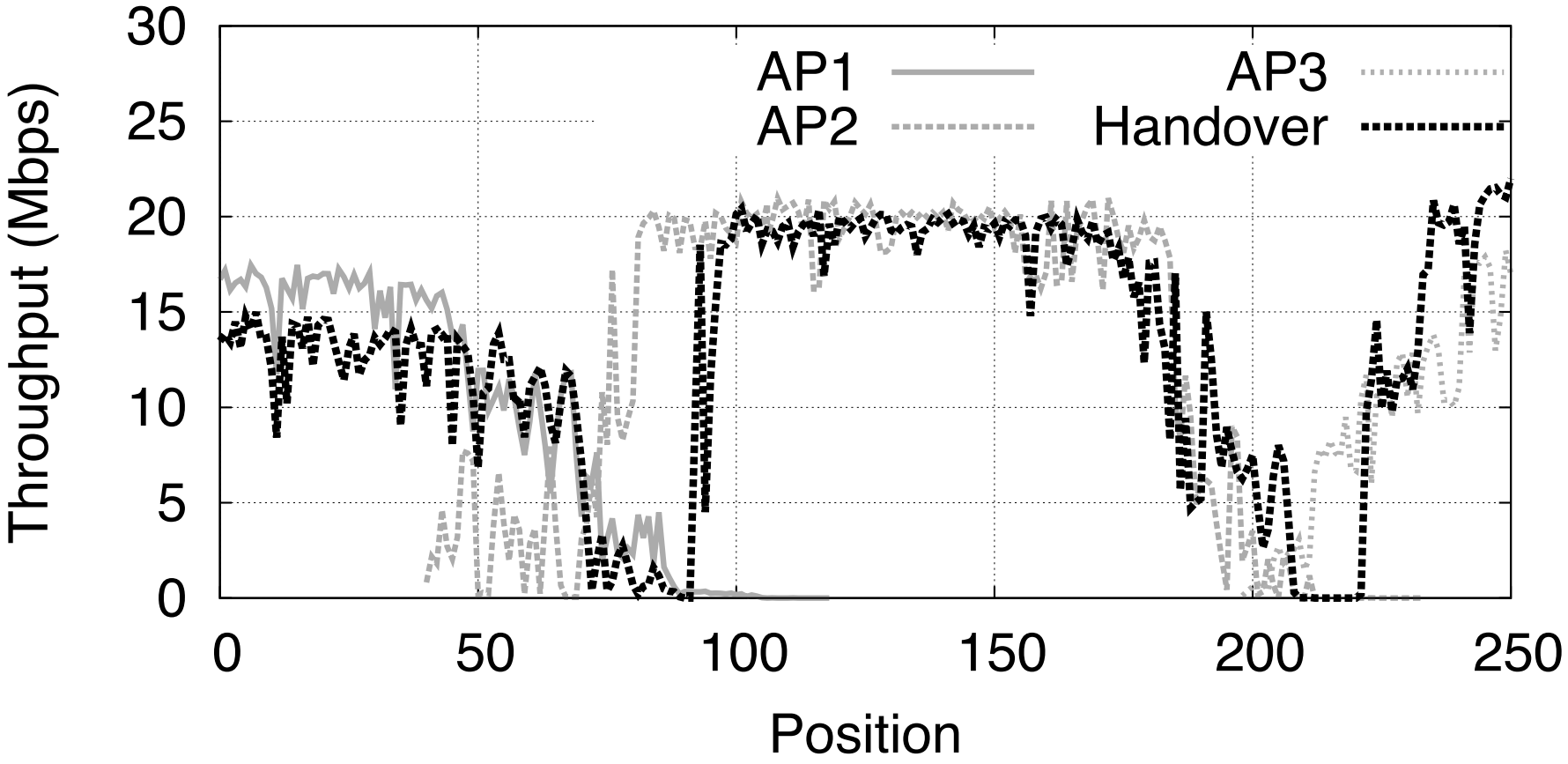




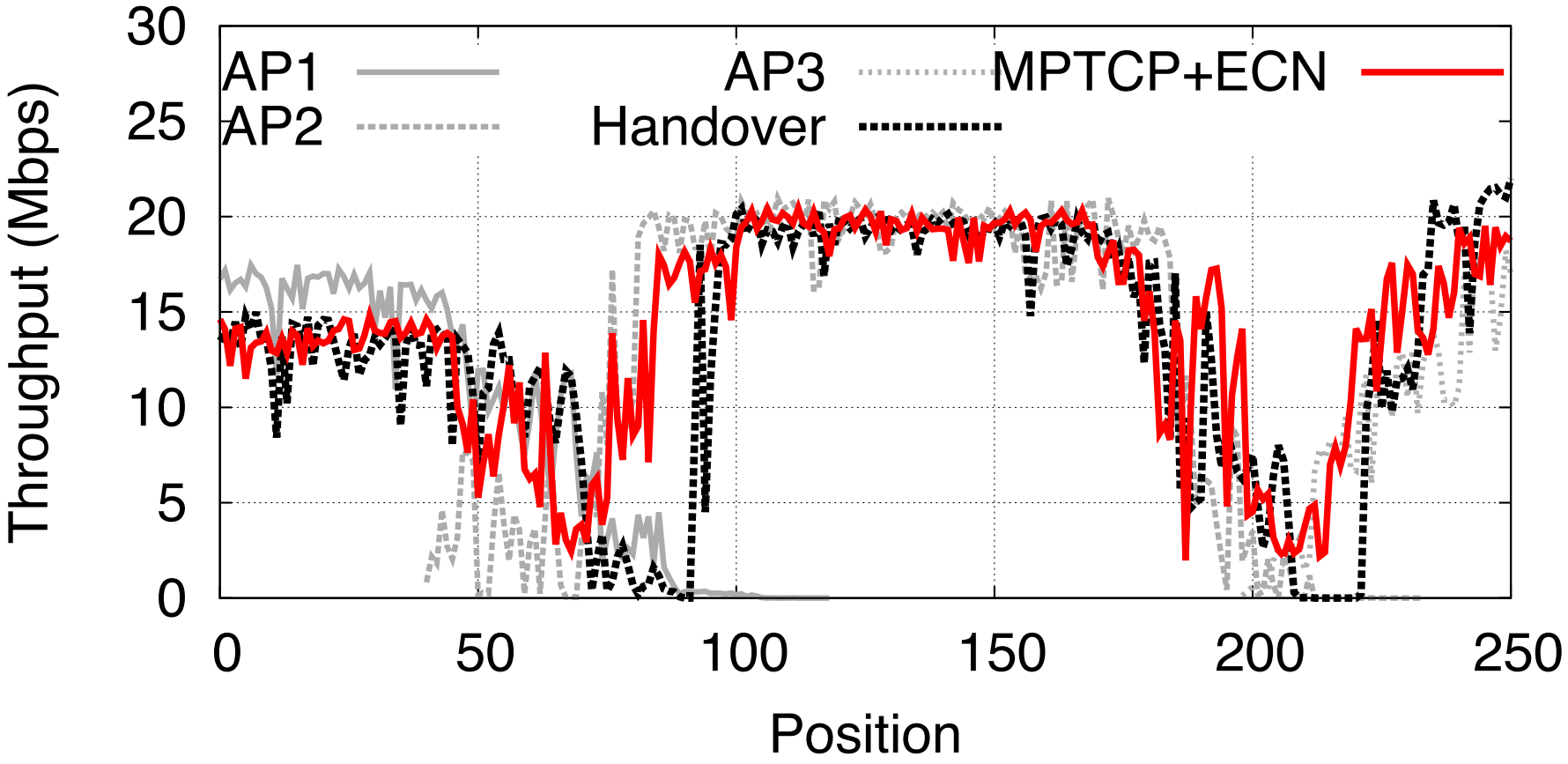
Indoor client, walking speed
A mobility experiment



Indoor client, walking speed
A mobility experiment



Indoor client, walking speed
A mobility experiment



Conclusions

MPTCP + WiFi = 

Match made in heaven:

- Hidden terminals
- CS, all senders use the same rate
- CS, minstrel

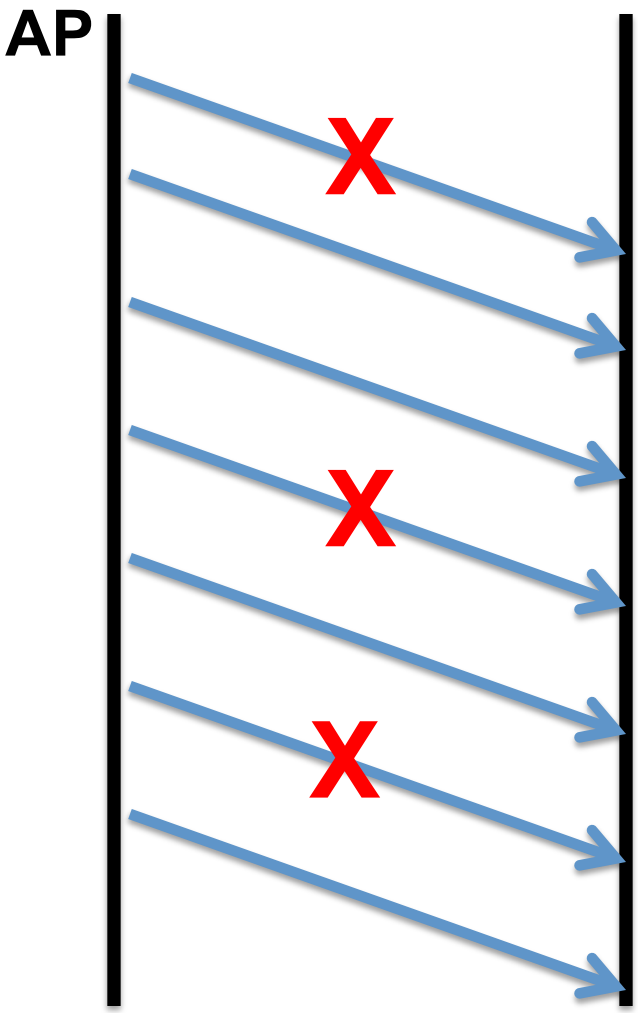
Human intervention needed via ECN:

- Robust across all cases we tested

Code available at: http://mobil4.org/multiwifi/

Backup

Estimating the PHY Loss rate

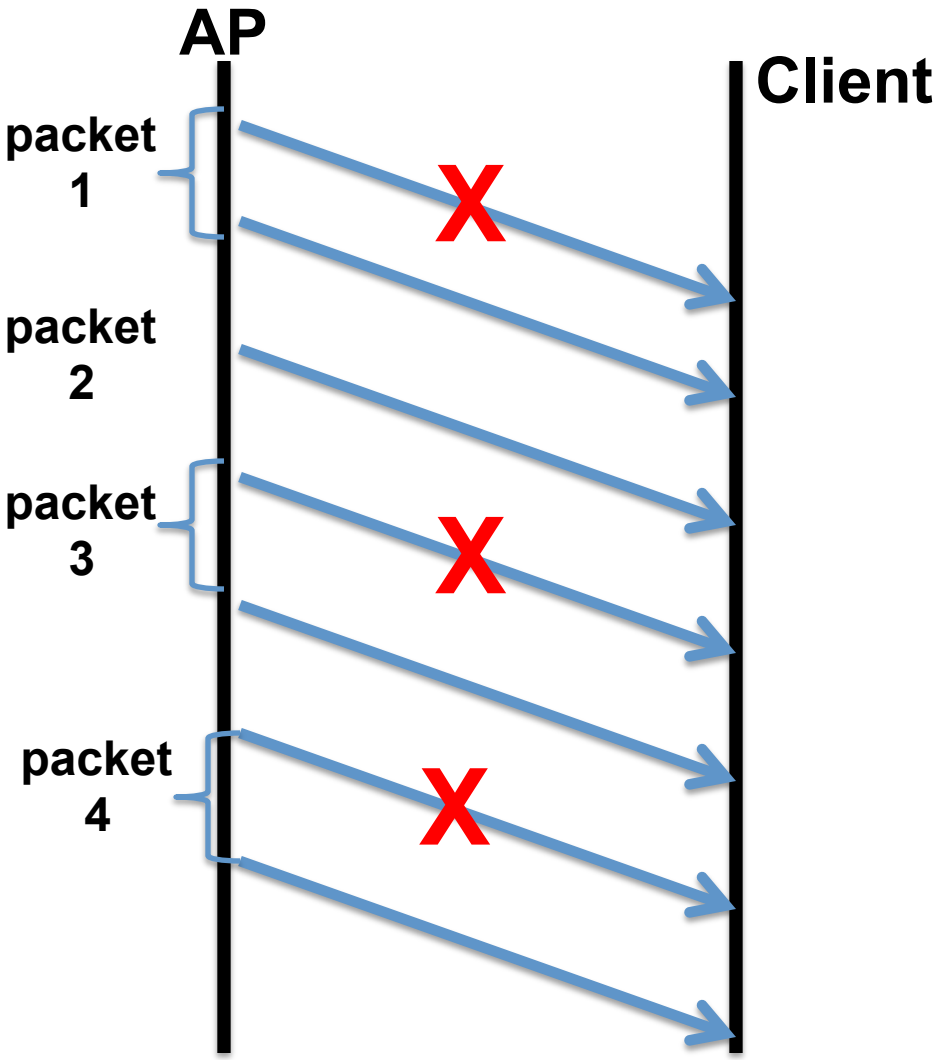


Obvious formula:

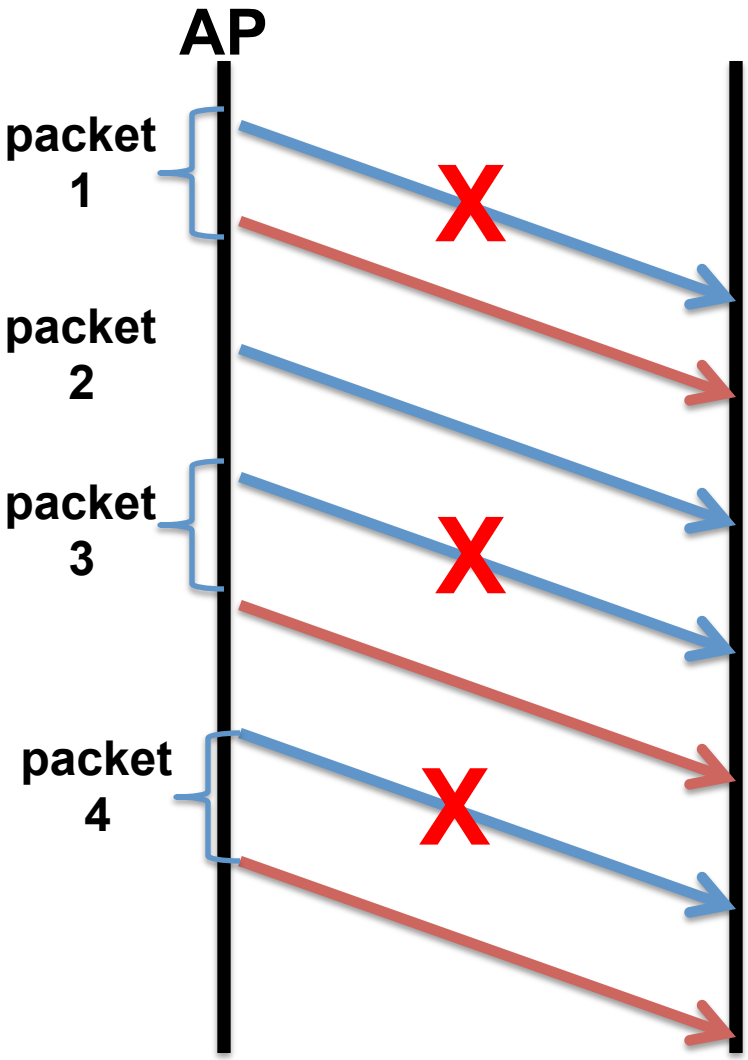
$$delivery_prob = \frac{N_{received}}{N_{total}}$$

Client doesn't know N_{total}

Estimating the PHY Loss rate: Retry bit



Estimating the PHY Loss rate: Retry bit



Estimate delivery probability for first transmission of each packet

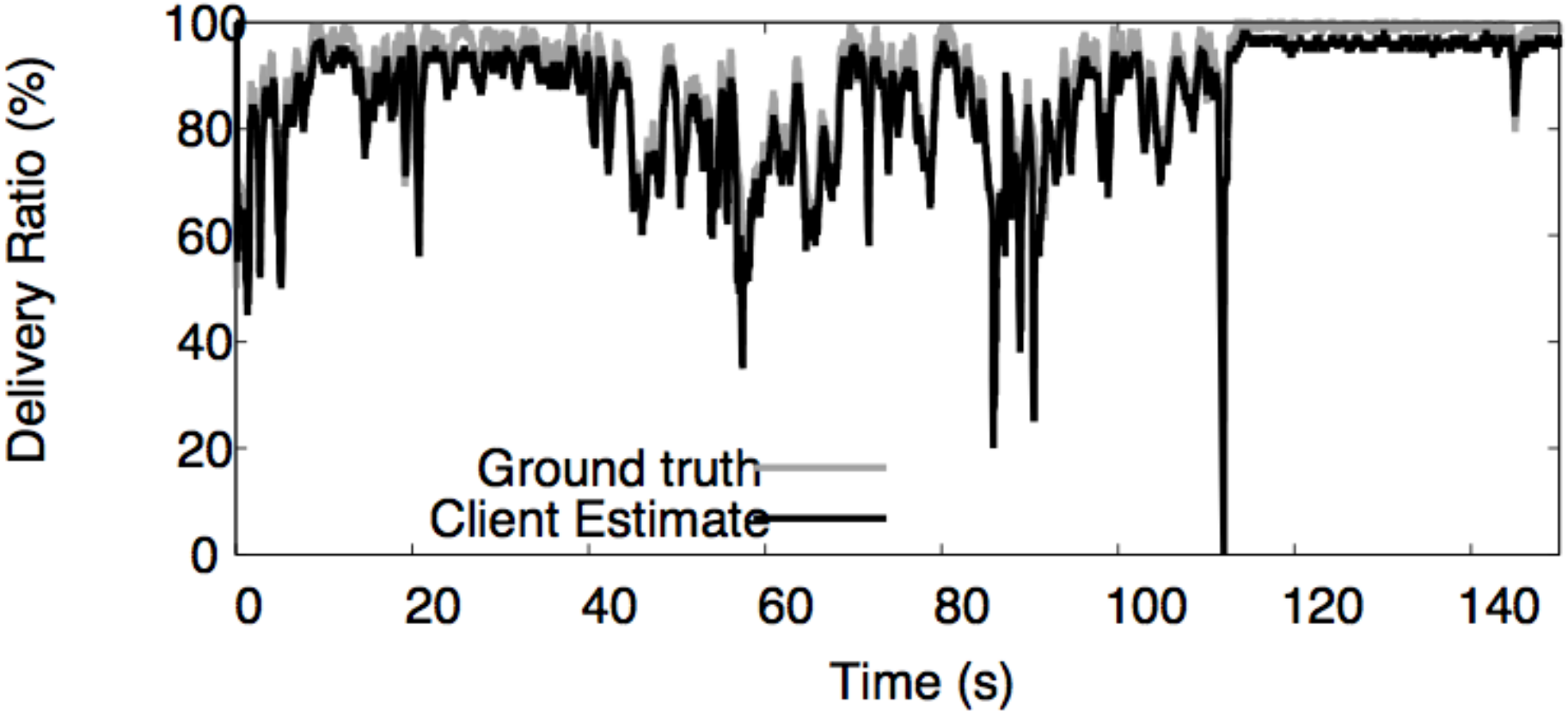
$$delivery_prob = \frac{N_0}{N_0 + N_1 + N_{lost}}$$

N_0 #packets received without retry bit=0

N_1 #packets received without retry bit=1

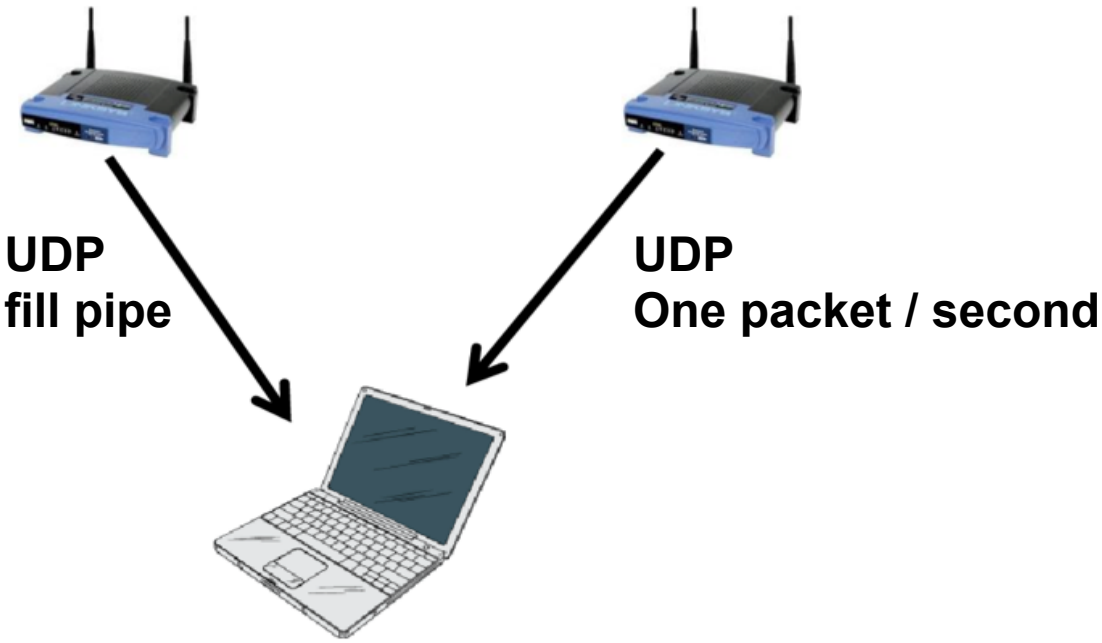
N_{lost} #lost packets

Client-side loss estimation in practice

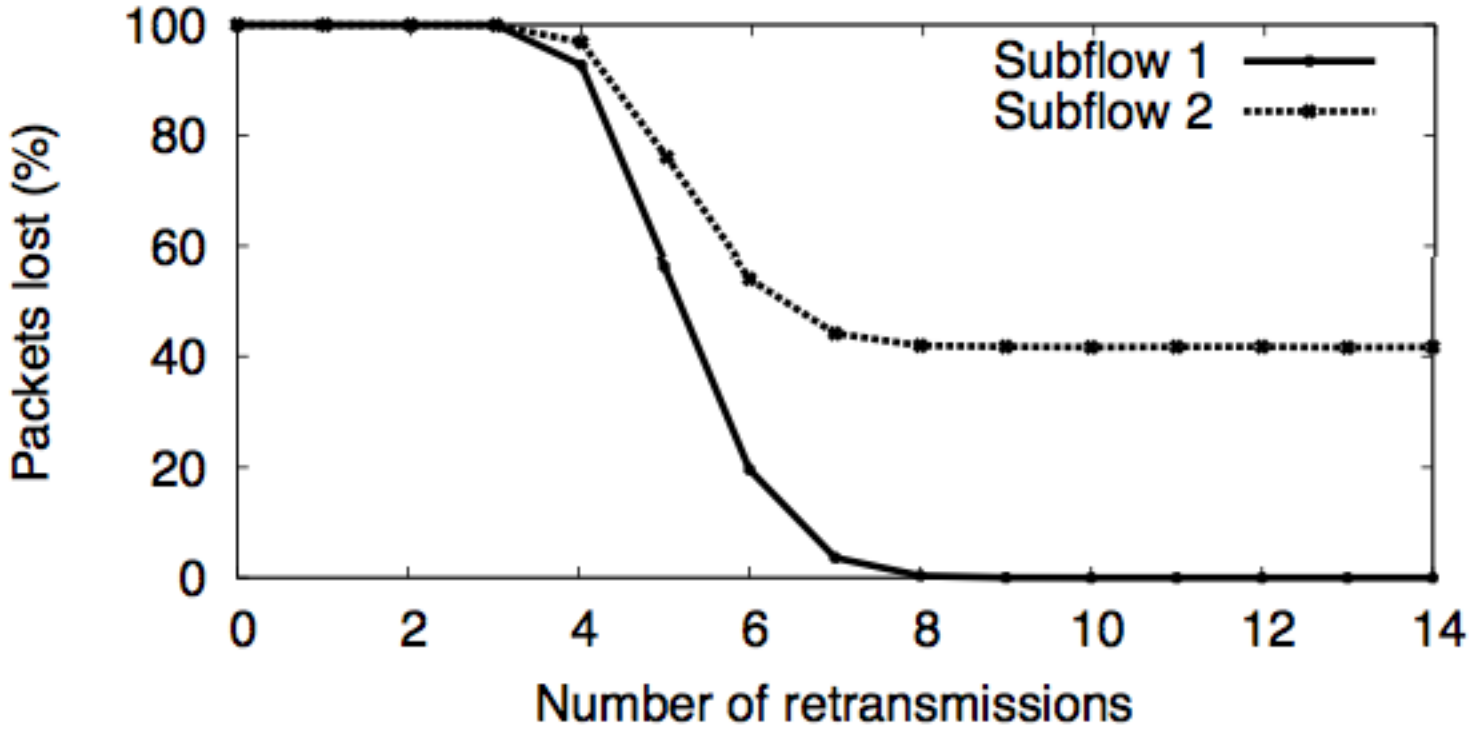


Simple experiment to understand what's going on

Ns2 simulation



Loss experienced as a function of Wifi Retry Count



Stationary client

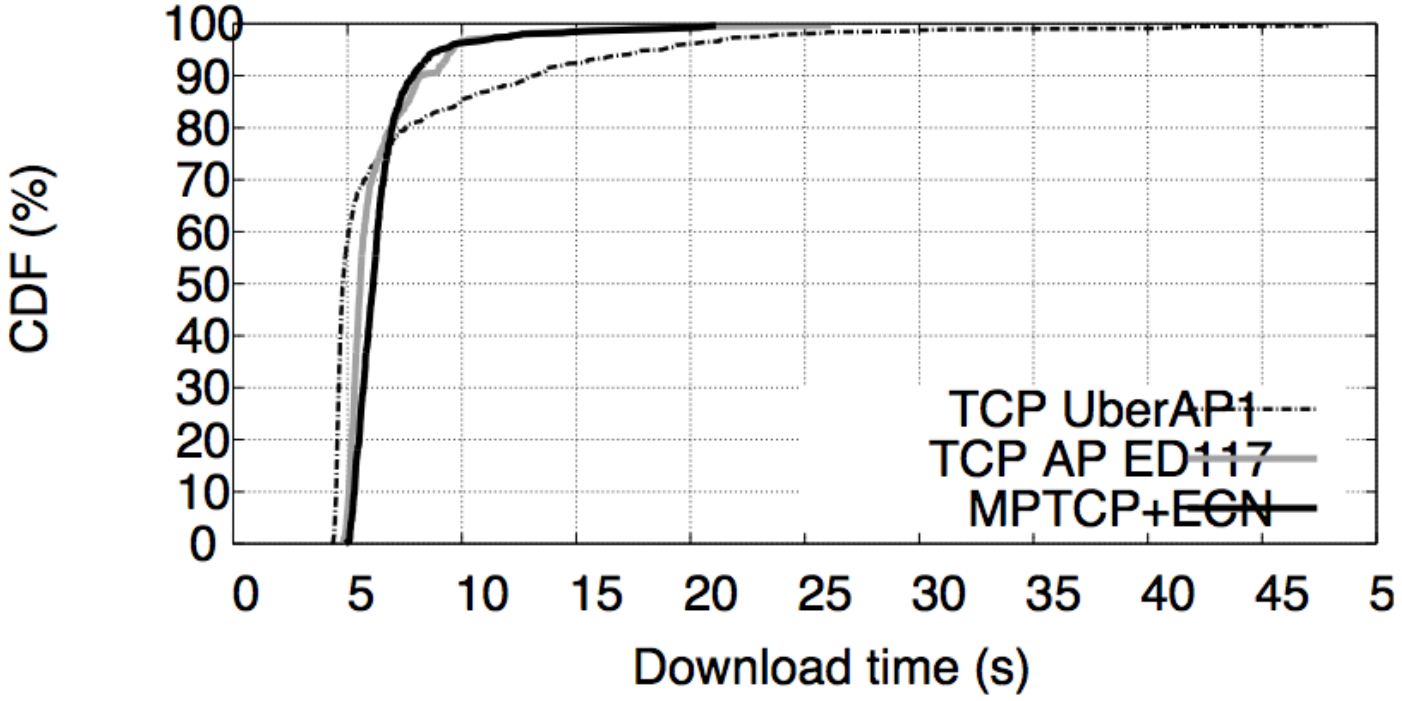


Figure 9: Static clients experience performance variations outside their control; MPTCP offers predictable performance