Three ways to (ab)use Multipath Congestion Control

Costin Raiciu

University Politehnica of Bucharest

Multipath TCP Design Goals

Do at least as well as TCP on any path

Move traffic away from congested subflows

Be fair to TCP at bottlenecks

How does TCP congestion control work?

Maintain a congestion window w.

- Increase w for each ACK, by $1/w$

- Decrease w for each drop, by $w/2$

How does MPTCP congestion control work?

Maintain a congestion window w_r, **one window for each path,** where $r \in R$ ranges over the set of available paths.

- Increase w_r for each ACK on path r, by $\displaystyle \min_{S \subseteq R : r \in S} \frac{\max_{s \in S} w_s / \mathrm{RTT}_s^2}{\left(\sum_{s \in S} w_s / \mathrm{RTT}_s\right)^2}$

How does MPTCP congestion control work?

Maintain a congestion window w_r, **one window for each path,** where $r \in R$ ranges over the set of available paths.

Design goal 3:
At any potential bottleneck S that path r might be in, look at the best that a single-path TCP could get, and compare to what I'm getting.

- Increase w_r for each ACK on path r, by

$$\min_{S \subseteq R : r \in S} \frac{\max_{s \in S} w_s / \mathrm{RTT}_s^2}{\left(\sum_{s \in S} w_s / \mathrm{RTT}_s\right)^2}$$

How does MPTCP congestion control work?

Maintain a congestion window wr, **one window for each path,** where r ∈ R ranges over the set of available paths.

Design goal 2:
We want to shift traffic away from congestion.

To achieve this, we increase windows in proportion to their size.

- Increase wr for each ACK on path r, by $\min_{S\subseteq R\,:\,r\in S} \frac{\max_{s\in S} w_s/\mathrm{RTT}_s^2}{\left(\sum_{s\in S} w_s/\mathrm{RTT}_s\right)^2}$

Multipath TCP Meets WiFi

AP deployment in Bucharest Center

WiFi is mostly a static connectivity solution

Why bother selecting best AP?

Connect to all APs

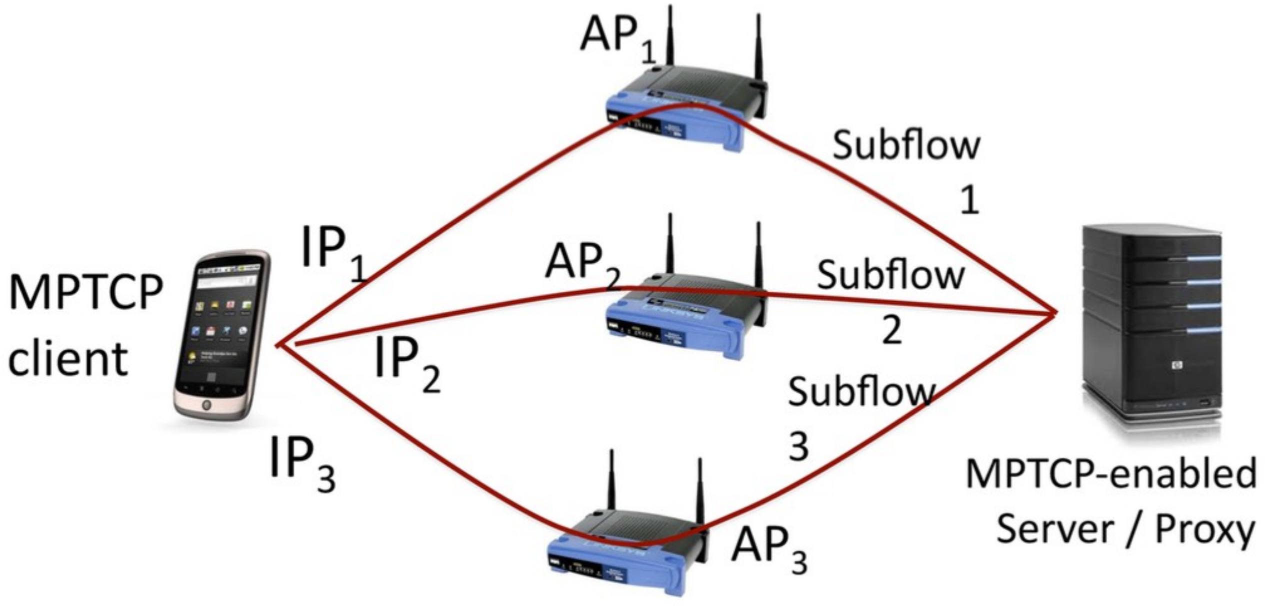
Spread traffic with Multipath TCP

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



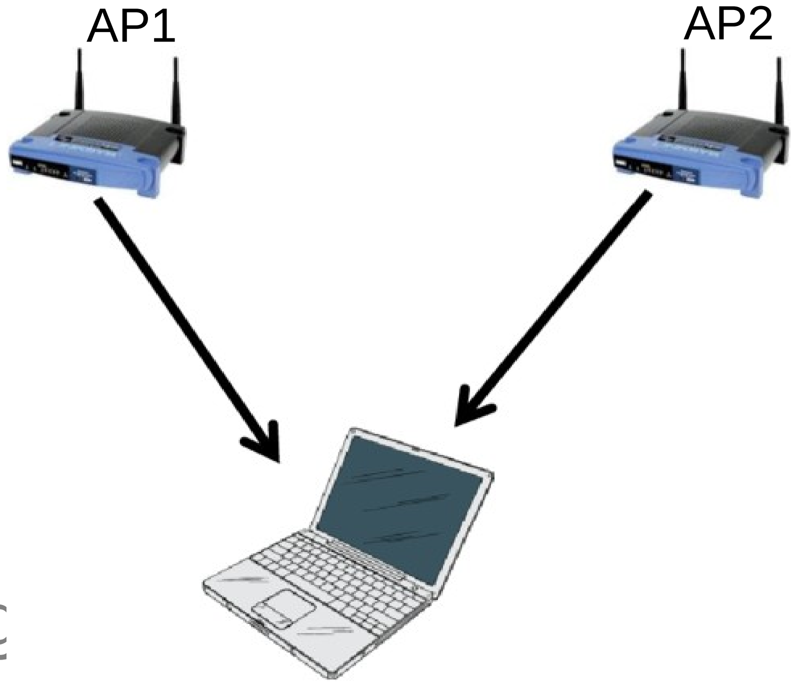
So how should traffic be allocated?

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

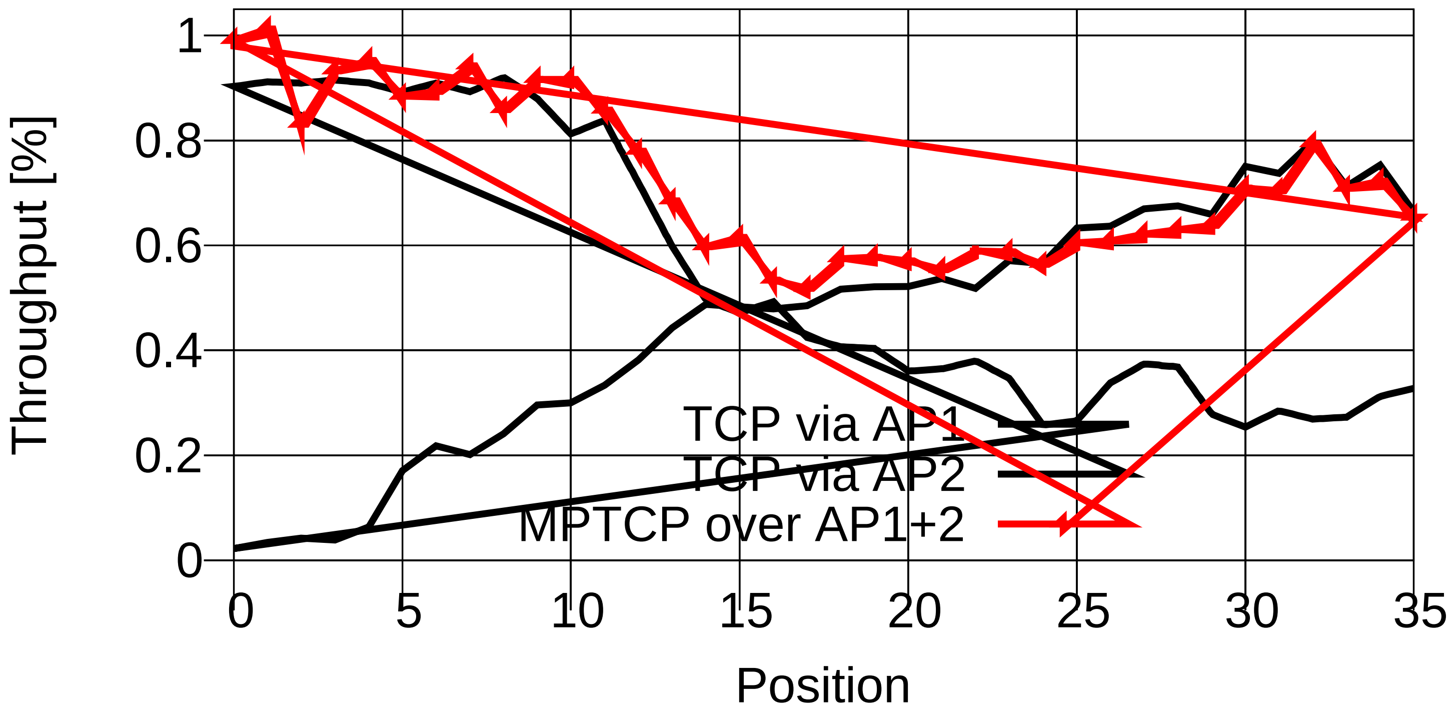
Multipath TCP + Multiple APs

Experiments

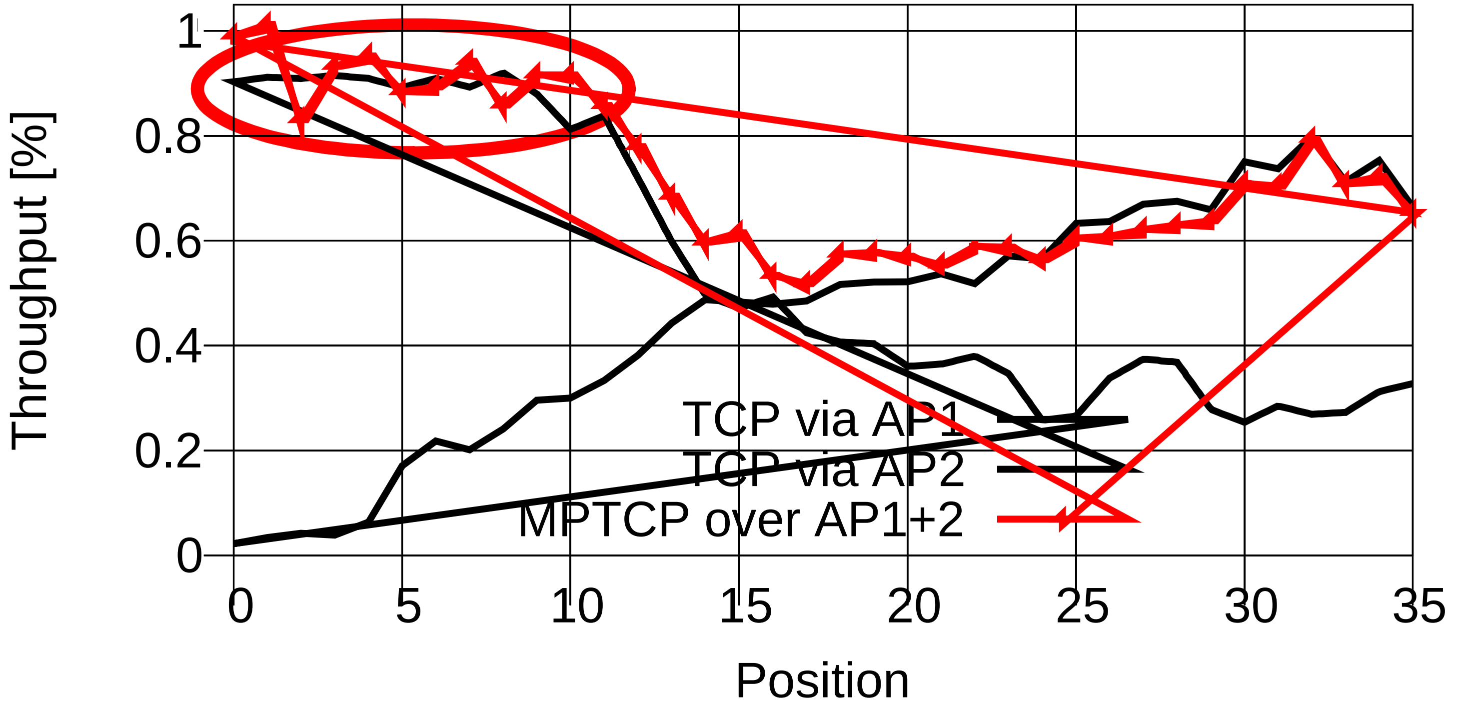
- Single channel
  - Hidden terminal
  - Carrier-sense
- Multiple channels
  - Use channel switching



Carrier Sense experiment
APs hear each other



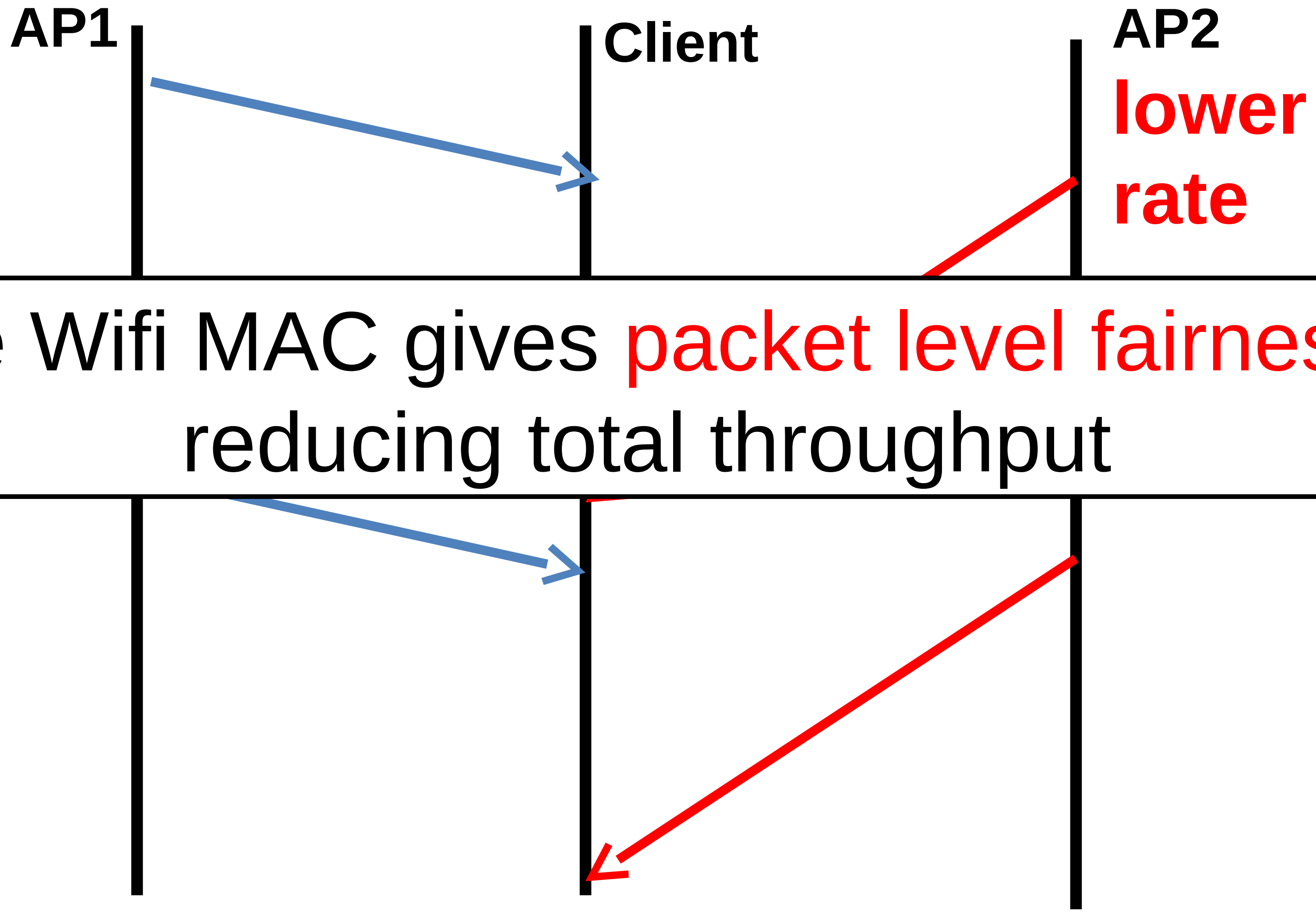
Carrier Sense experiment
APs hear each other



Great. Are we done? **NO**

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

MAC with two APs sending



The Wifi MAC gives packet level fairness, reducing total throughput

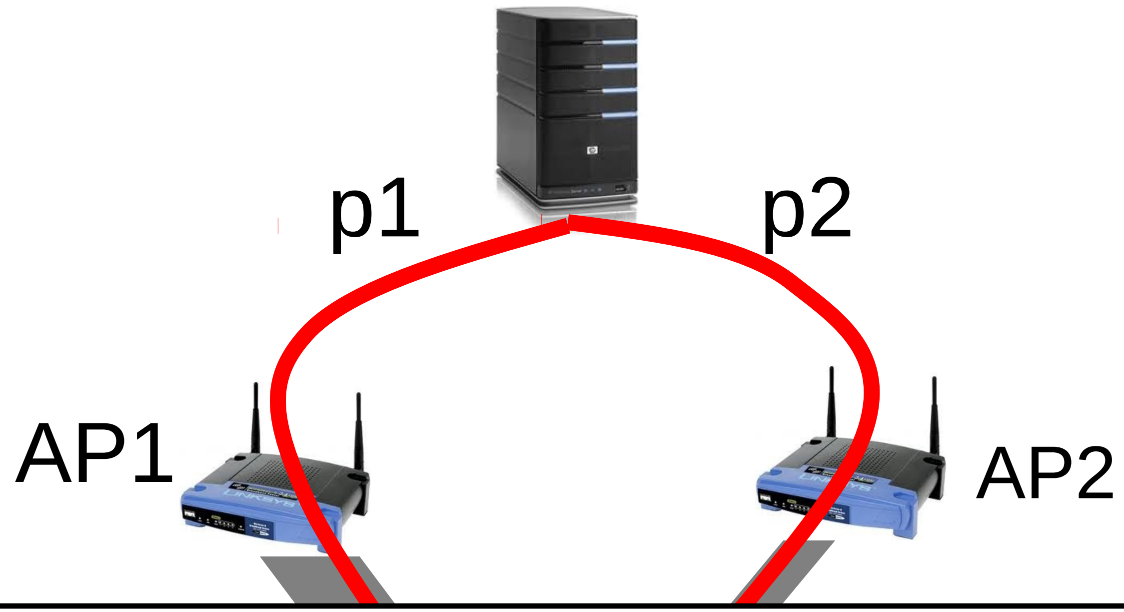
Carrier-Sense performance depends on rate control algorithm

- Near-optimal behaviour when:
  - All APs used the same fixed rate

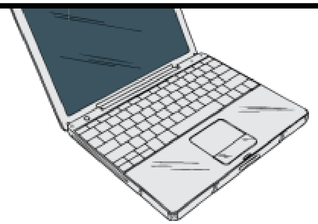
Rate control algorithms are specific to each AP vendor. Change is very difficult.

- Bad behaviour when:
  - Using less aggressive rate control algorithms
  - Using *minstrel-ht* (802.11n)

MPTCP congestion control in WiFi



When the MAC gives packet level fairness, $p1 = p2$



Client knows which AP is better

Client estimates the average time Ti it takes APi to send one packet

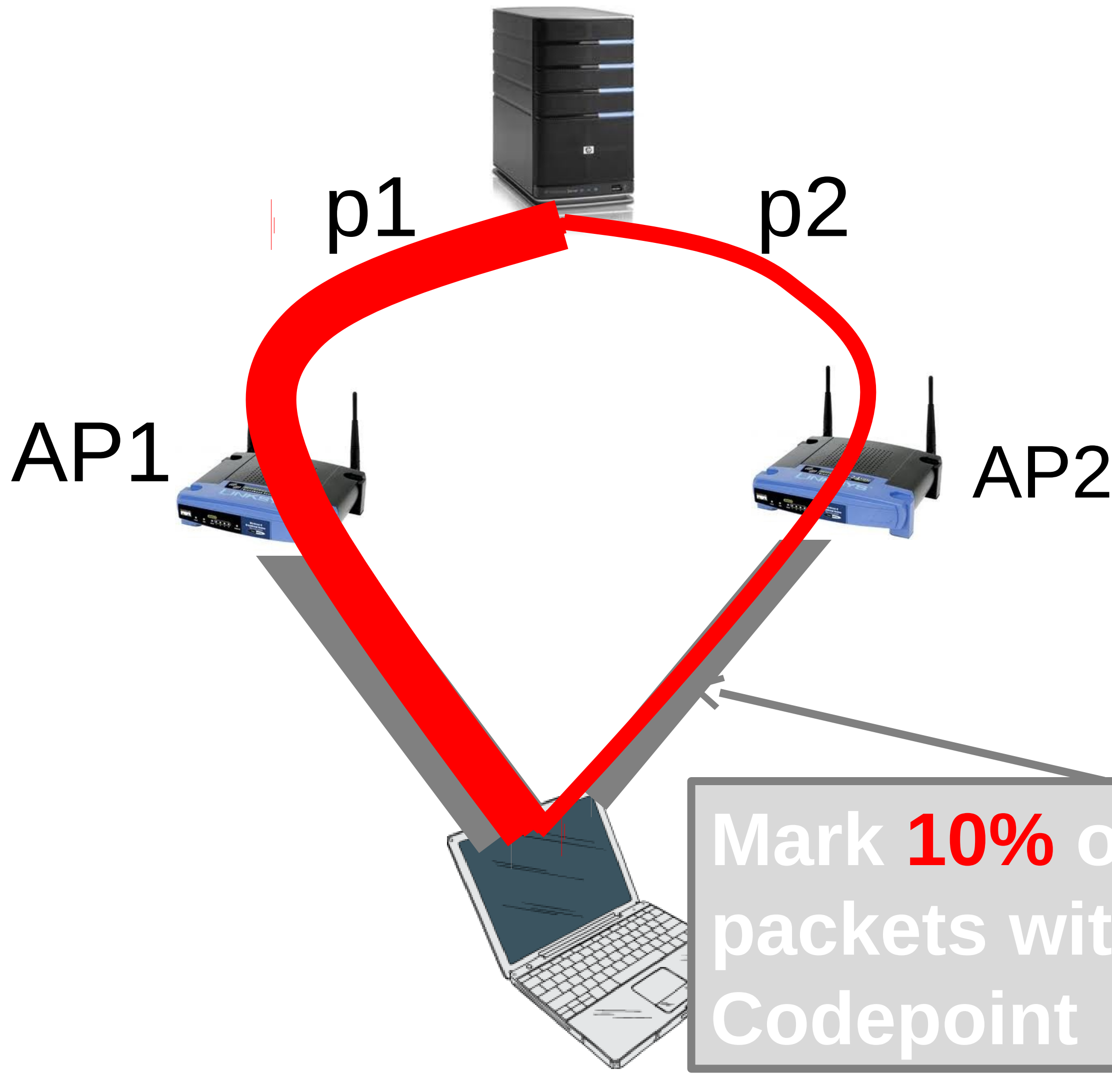
- Assuming AP is alone accessing the medium
- Passively estimate PHY loss rates [see paper]

Client orders its APs according to Ti

- Informs server that APj is bad when

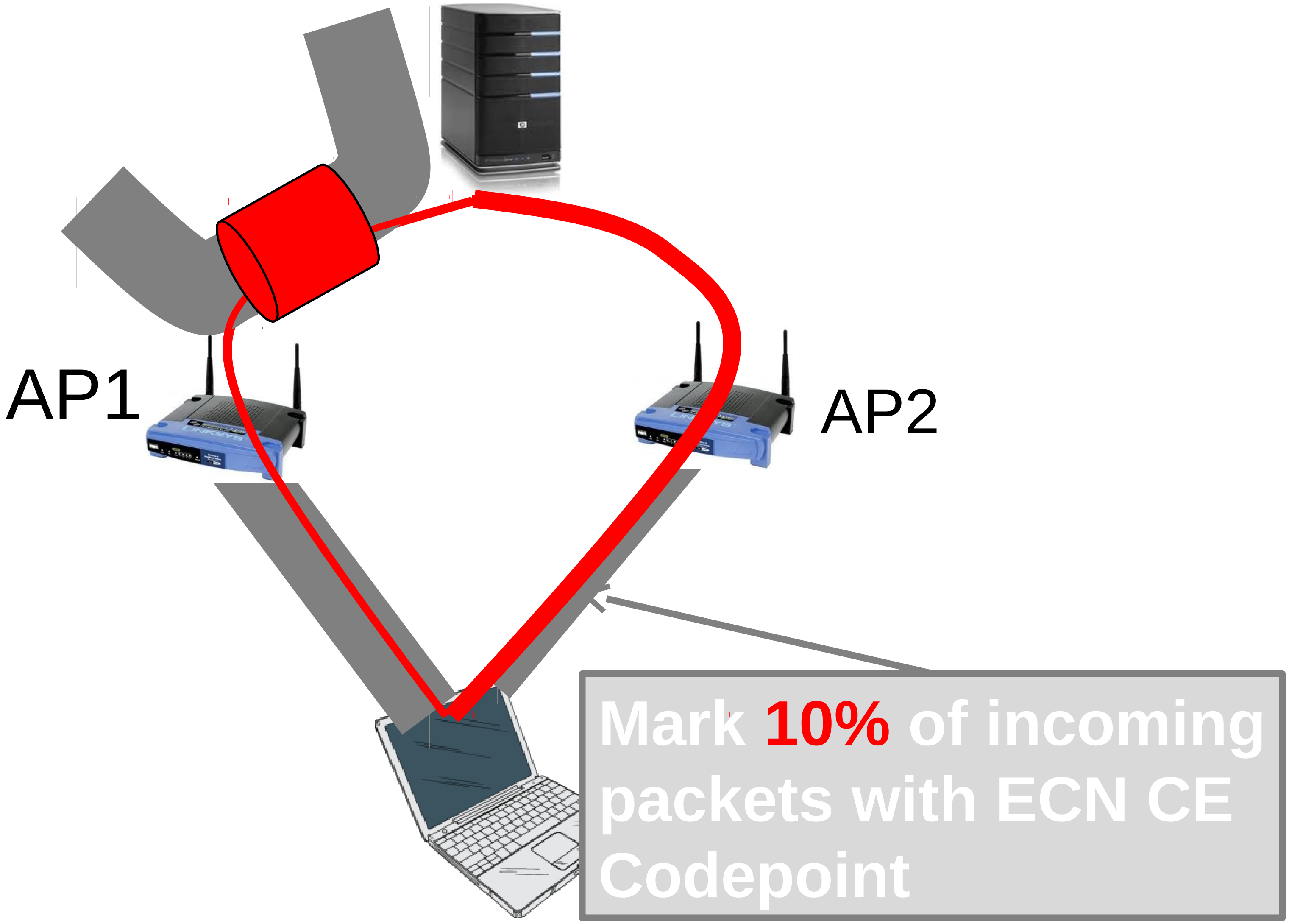
  Tj > 1.3 mini=1,N(Ti)

Client ECN marking to steer traffic



Mark **10%** of incoming packets with ECN CE Codepoint

Client ECN marking to steer traffic



MPTCP congestion control primer

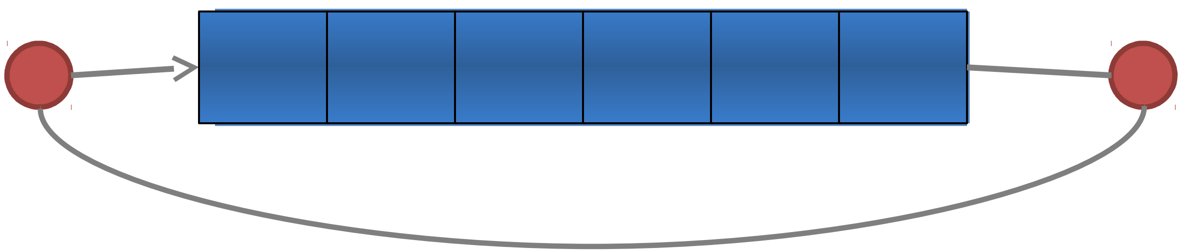
Move traffic away from congested links

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

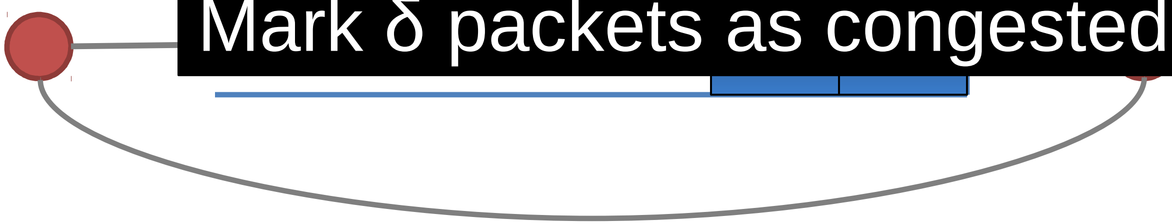
Do at least as good as TCP on the best path

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

Safe ECN Marking



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



Mark δ packets as congested using ECN

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

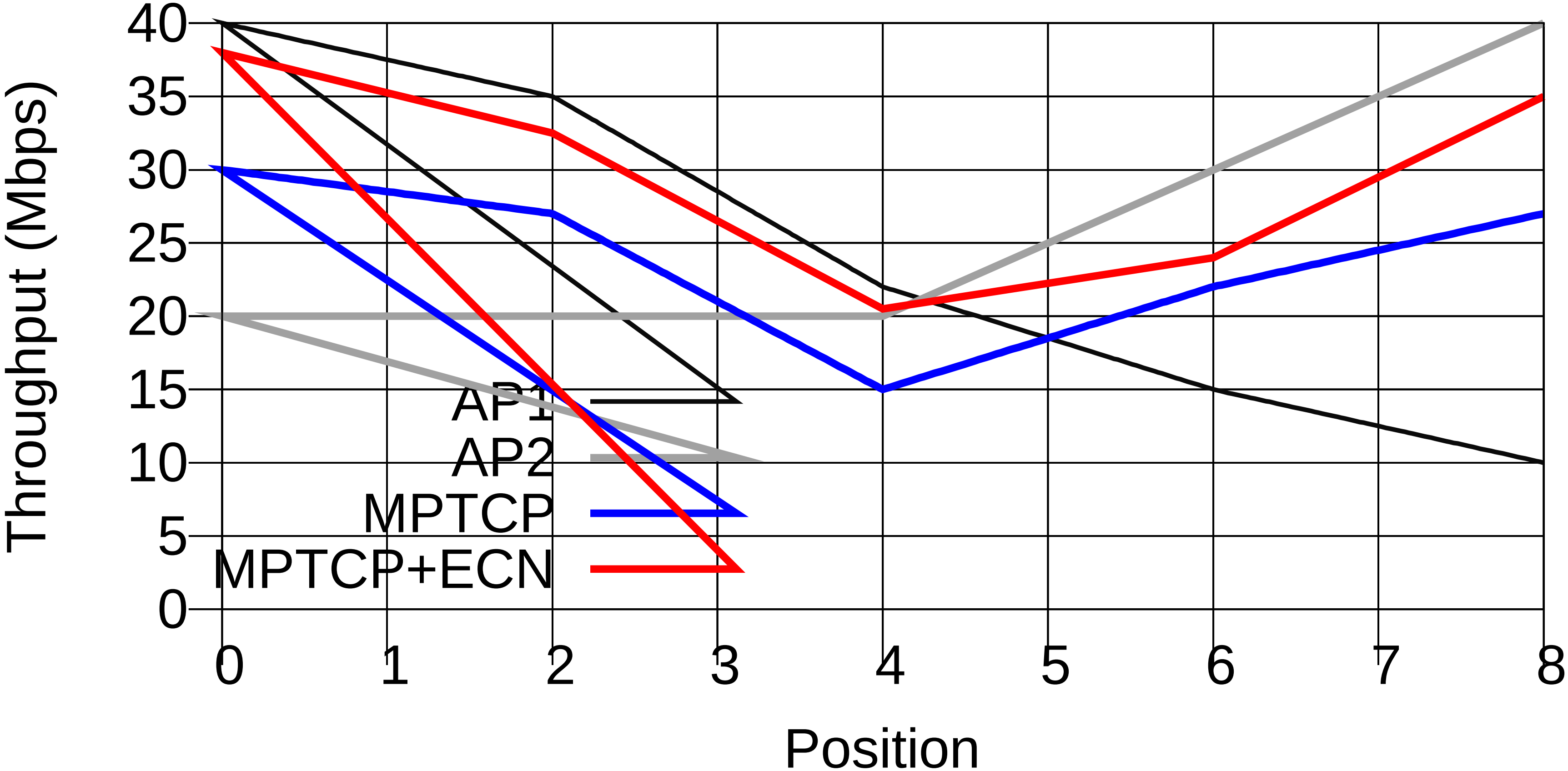
Safe ECN Marking

The client can compute the safe marking threshold delta:

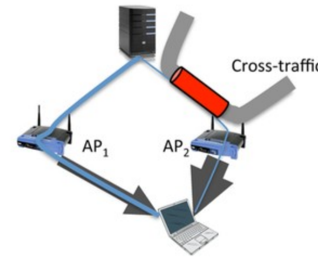
$$\delta = \frac{1}{2} \cdot \left(\frac{50 \cdot T_1^2}{RTT \cdot (RTT - 50 \cdot T_1)}\right)^2$$

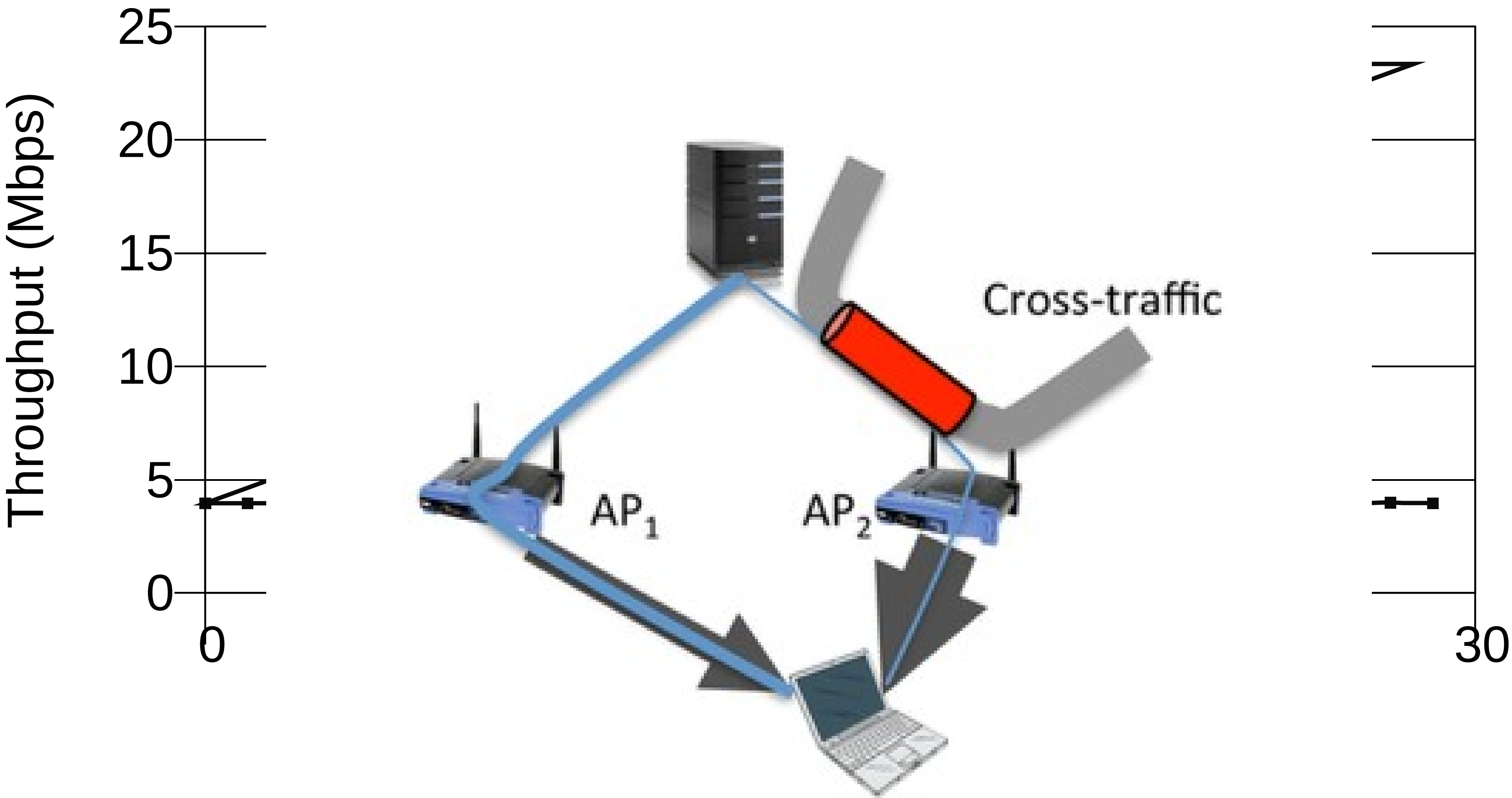
Implemented in device-independent part of the WiFi driver of the Linux kernel.

How well does ECN marking work for 802.11n?

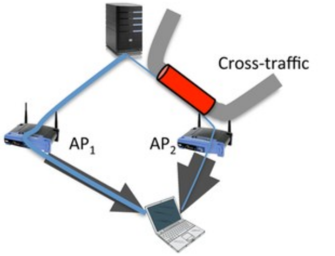


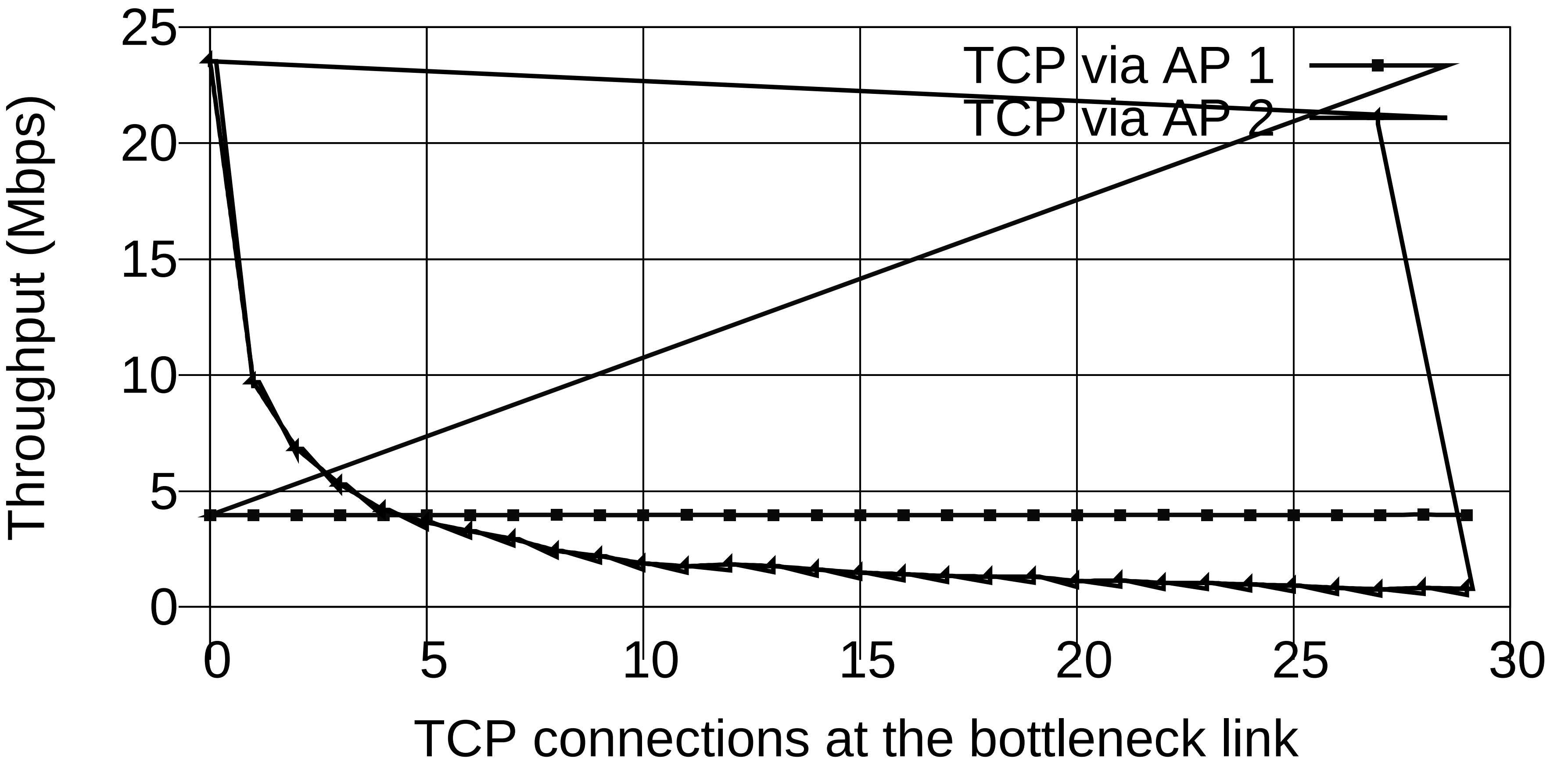
Ensuring good performan with bottlenecked APs



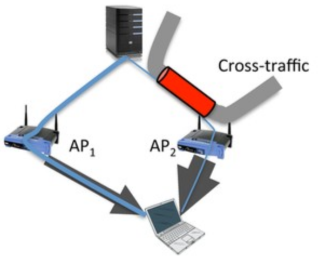


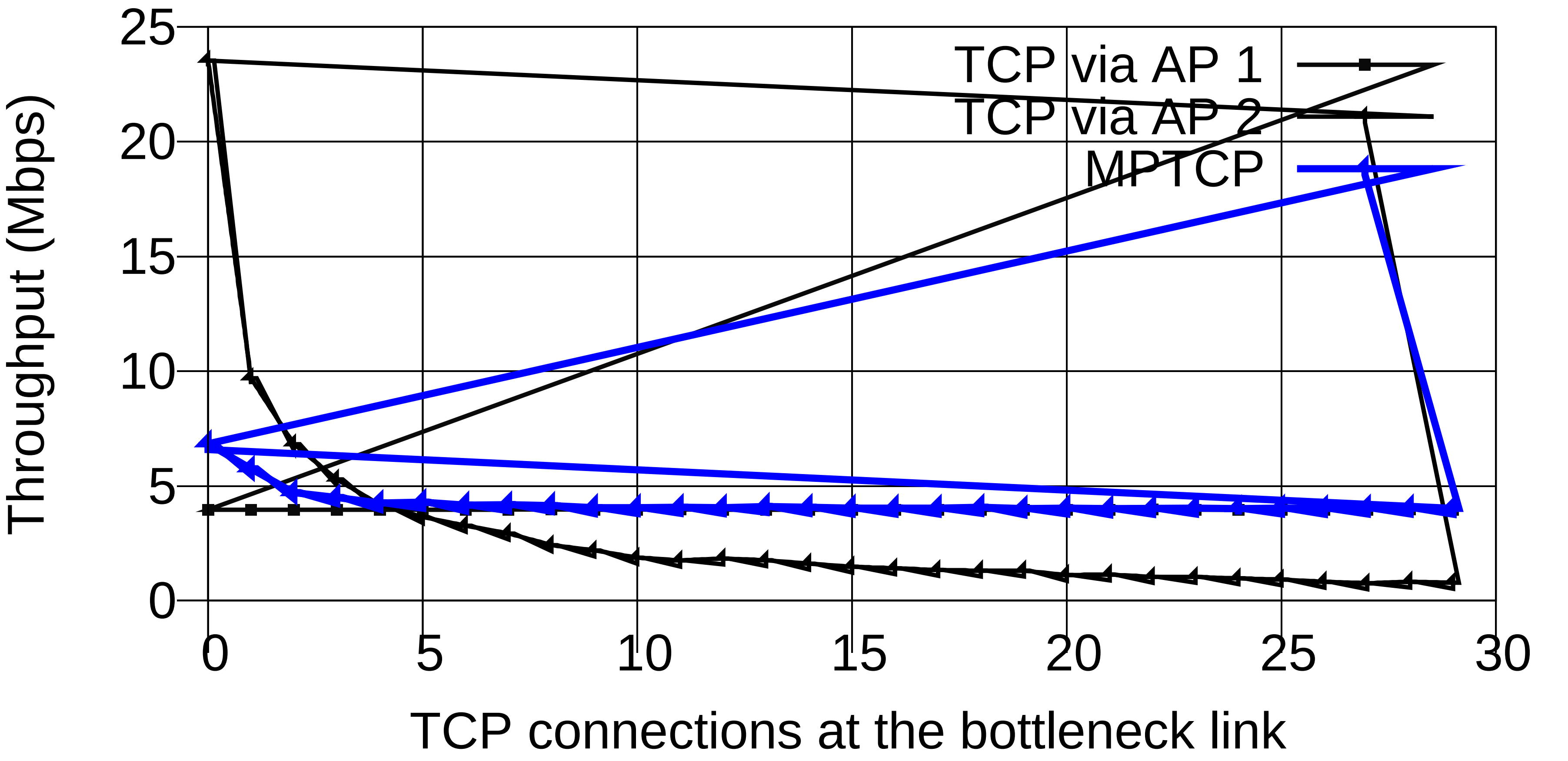
Ensuring good performan with bottlenecked APs



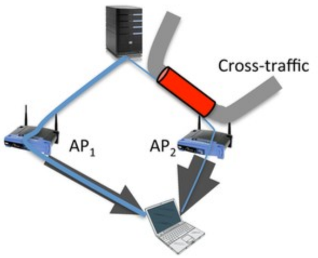


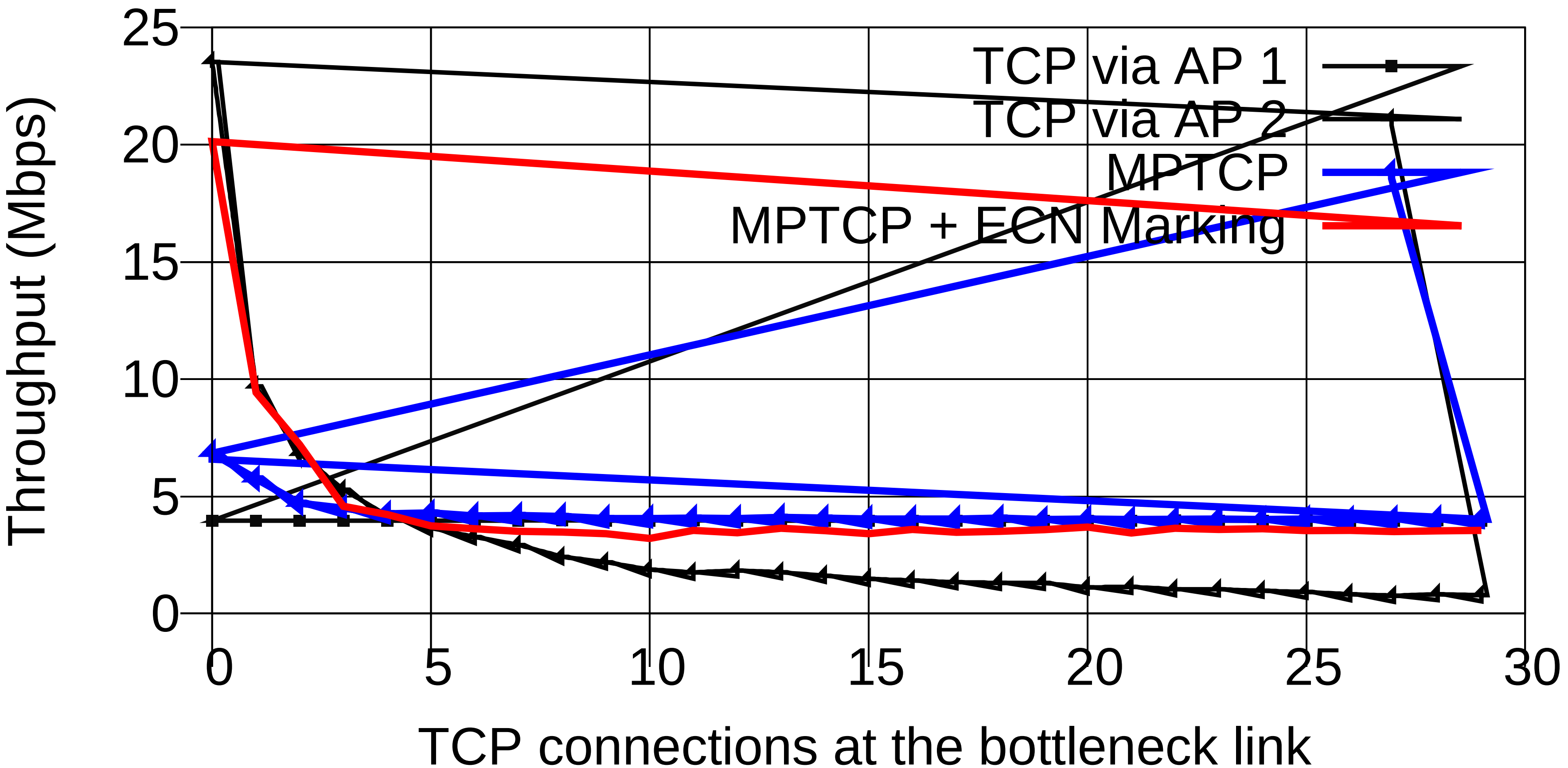
Ensuring good performan with bottlenecked APs



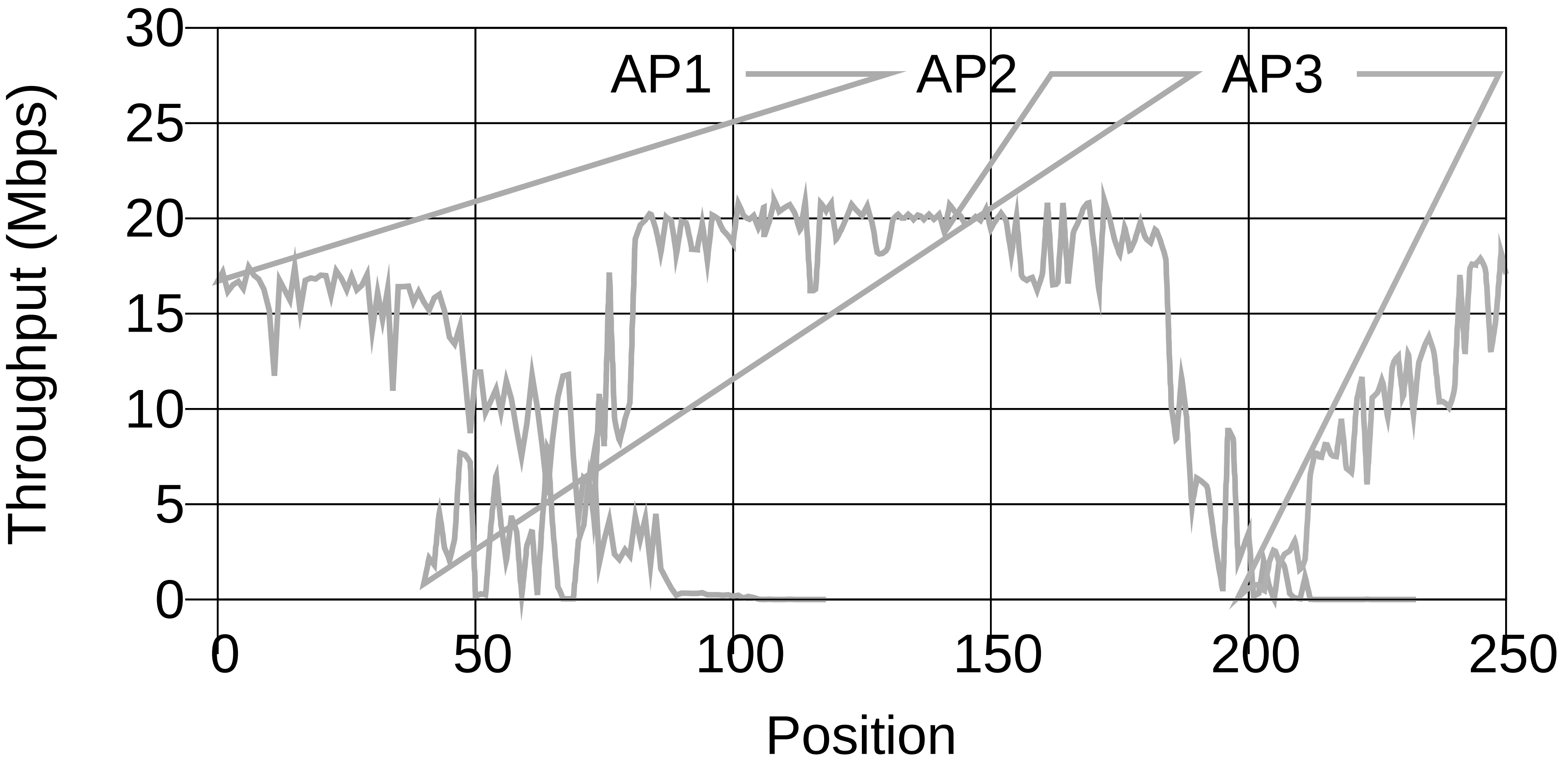


Ensuring good performan with bottlenecked APs

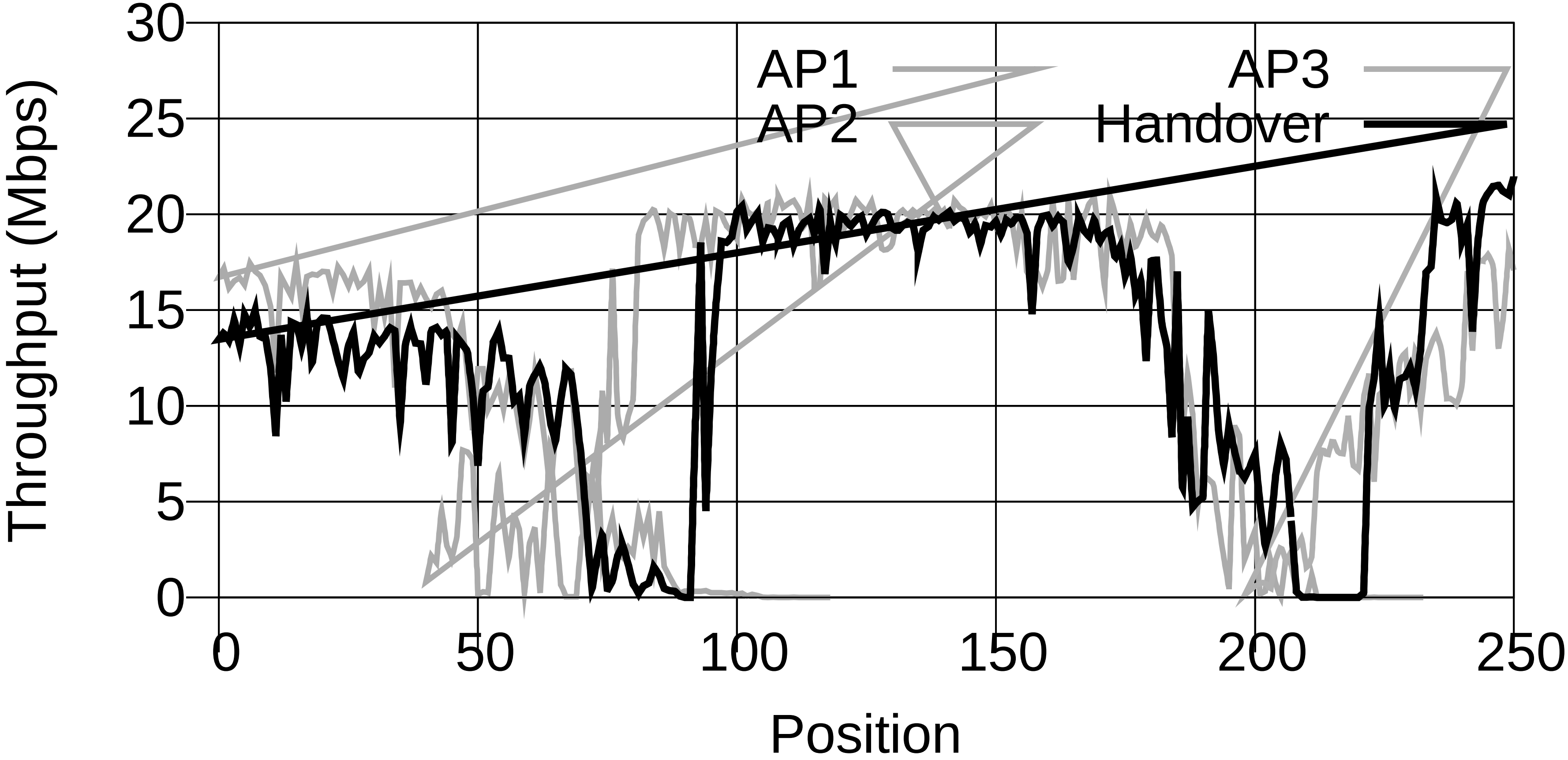




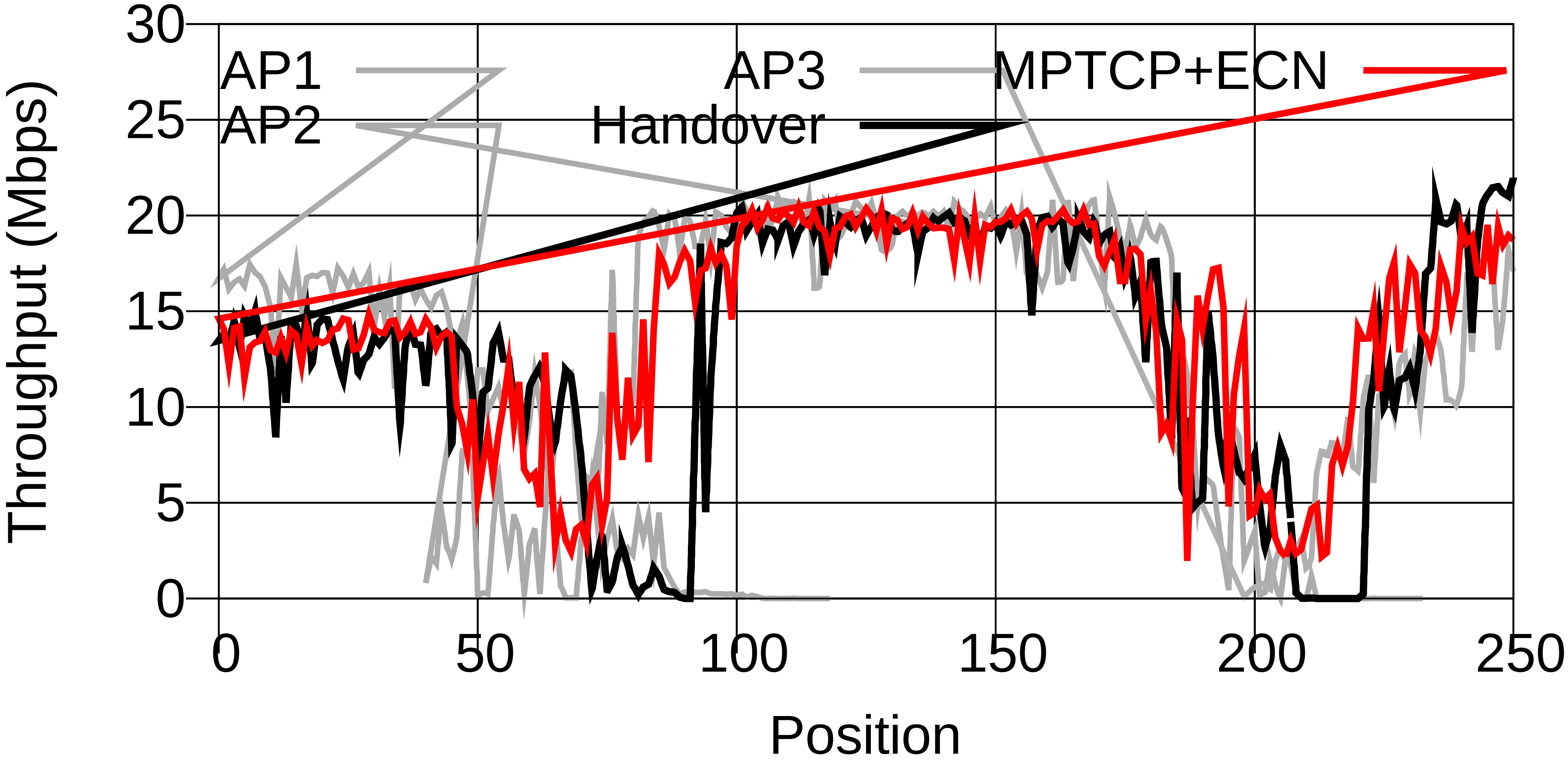
Indoor client, walking speed
A mobility experiment



Indoor client, walking speed
A mobility experiment

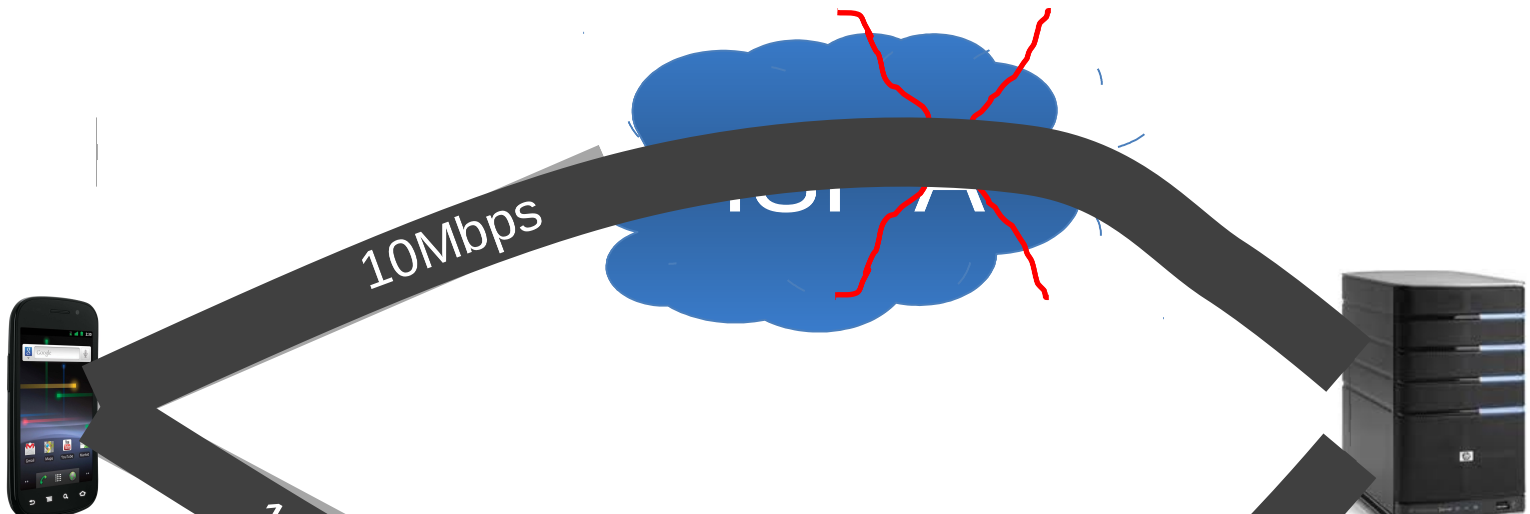


Indoor client, walking speed
A mobility experiment



Internet Operator Games with Multipath TCP

When does Multipath TCP give throughput benefits?

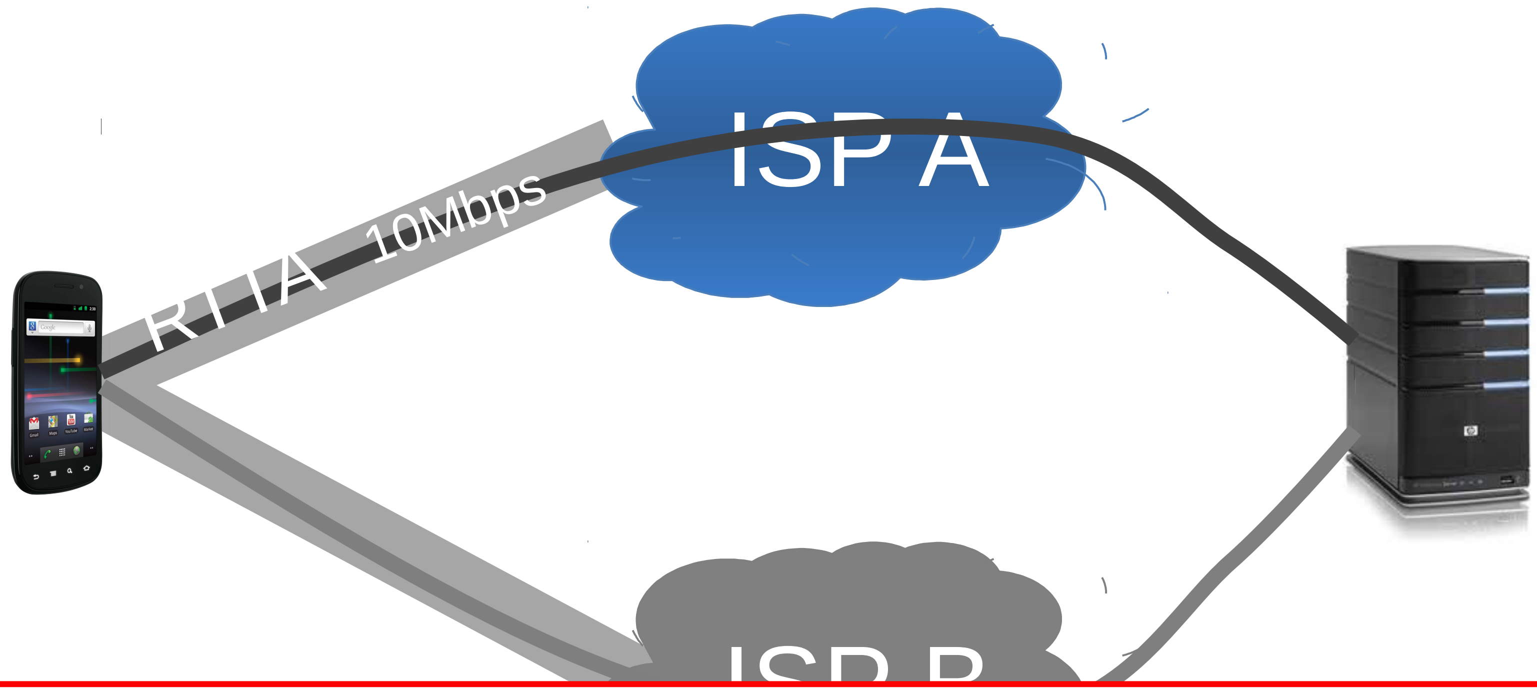


CASE 1: EXOGENEOUS LOSS

MPTCP flow does not affect loss rates

MPTCP = MAX (TCPA , TCPB)

Can ISPs game Multipath TCP clients?



fraction

Drop rate that does not affect TCP

The provider can easily compute the safe drop rate delta:

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

Where T1=1/Bclient ; RTT can be estimated cheaply.

Provider shaping at work

Shaping?	Provider A	Provider B	Total (Mbps)
	10	10	20
A	1	10	11
A + B	5.5	5.5	11

Provider shaping at work

TCP

Shaping?	Provider A	Provider B	Total (Mbps)
	10	10	20
A	9.5	10	19.37
A + B	9.5	9.3	18.8

Operator games

- Operators have incentives to shape
  - Short term bandwidth reduction offsets costs in shaping hardware
  - In the long run, everyone will shape.
- Using uncoupled congestion control fixes the problem

Ninja Tunneling

Why doesn’t Skype work over my 3G connection?

Unpredictability leads to inefficiency

- Middleboxes make Internet unpredictable
  - Connectivity depends on protocols, port numbers and even payload
  - Packets may be changed arbitrarily in flight

- *New apps **must** act like old ones*➔tunneling is norm
  - Tradeoff between efficiency and reachability
  - Most apps choose reachability

Ninja tunneling

Clients use multiple tunnels <u>simultaneously</u> to send traffic

- *Tunnels include native IP, UDP, TCP, HTTP/HTTPS, DNS, covert channels and can change dynamically*
- *Traffic spread over the different tunnels with MPTCP*
- *Drop tunnels that where packet changes are detected*

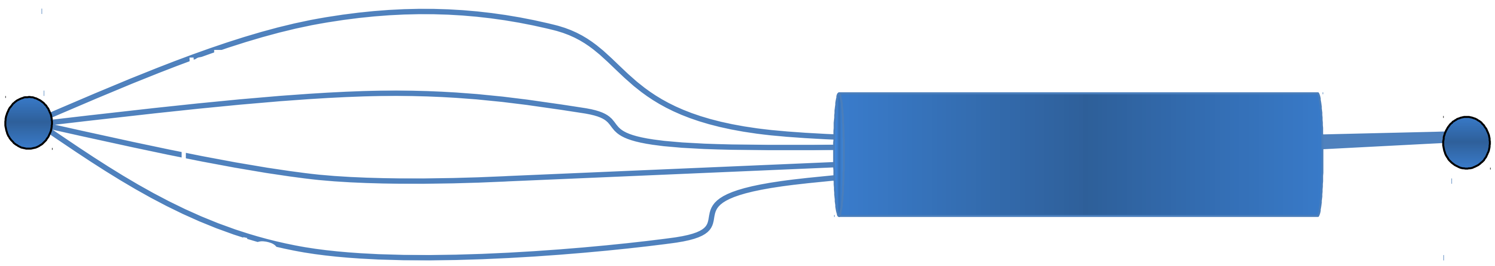
Ninja tunneling

Benefits

- *Unchanged apps*
- *Reachability is ensured in all cases*

***Key difficulty**: ensuring most traffic flows over the most efficient tunnel, and MPTCP congestion control can help*

Ninja tunneling in a nutshell



Multipath congestion control can be exploited by third parties to guide sender congestion control. It benefits:

- Clients:
  - In Wifi mobility
  - To circumvent operator DPI
- Operators:
  - To shed multipath traffic to other providers
  - To load balance their networks

Reduced buffer usage for TCP as a side-effect

Marking is harmful when multiple parties use