

UCast: Improving WiFi Multicast

Jayashree Subramanian, Robert Morris
and Hari Balakrishnan

Latest Trends in Mobile Video+WiFi Networks

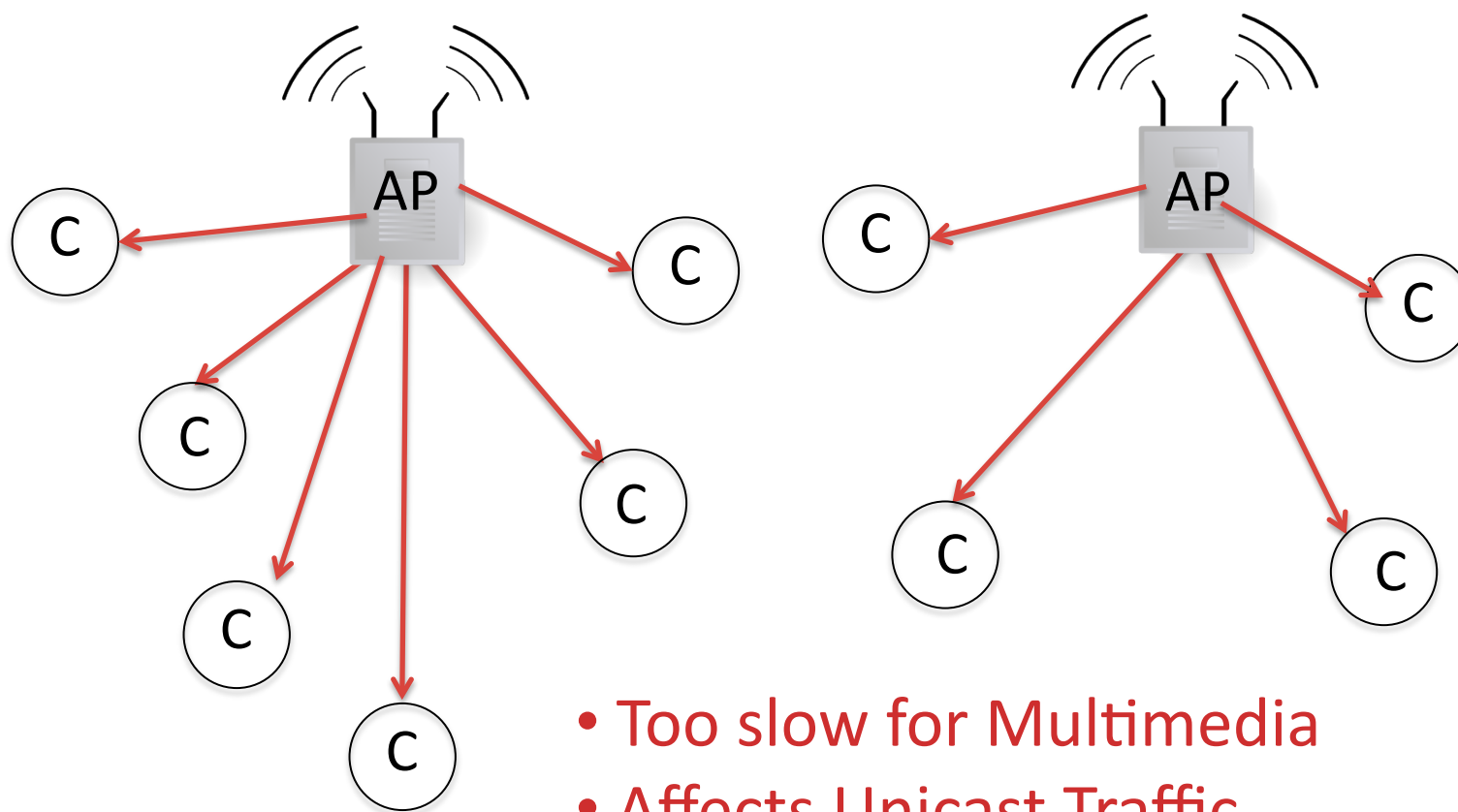
- More than 1 billion electronic devices with embedded WiFi chips by 2012
- By 2015, mobile video will generate 66% of all mobile traffic

WiFi Multicast Applications:

- Live video seminars and lectures in campuses and companies
- Live streaming services over metro-scale WiFi AP networks under single governance
 - City of Taipei has 2300 APs covering 50% of population

Traditional WiFi Multicasting

- Clients connect to the AP with highest RSSI
- Multicast → Unicast packets



- Too slow for Multimedia
- Affects Unicast Traffic

Key Ideas Behind UCast

1. Cooperative client multicasting

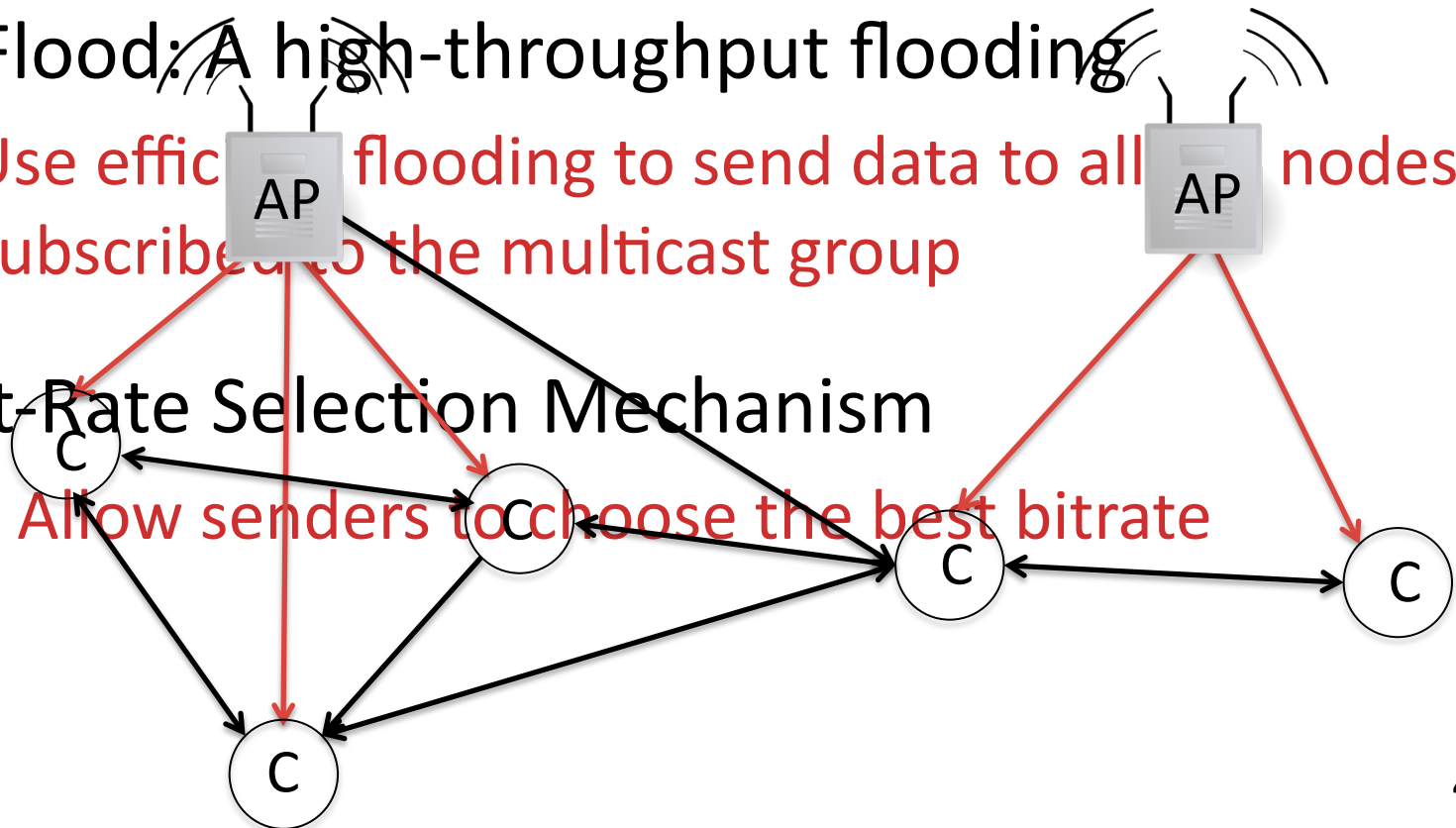
- Client forward on behalf of APs
- Talk to other APs
- Clients form a mesh network and flood packets

2. UFlood: A high-throughput flooding

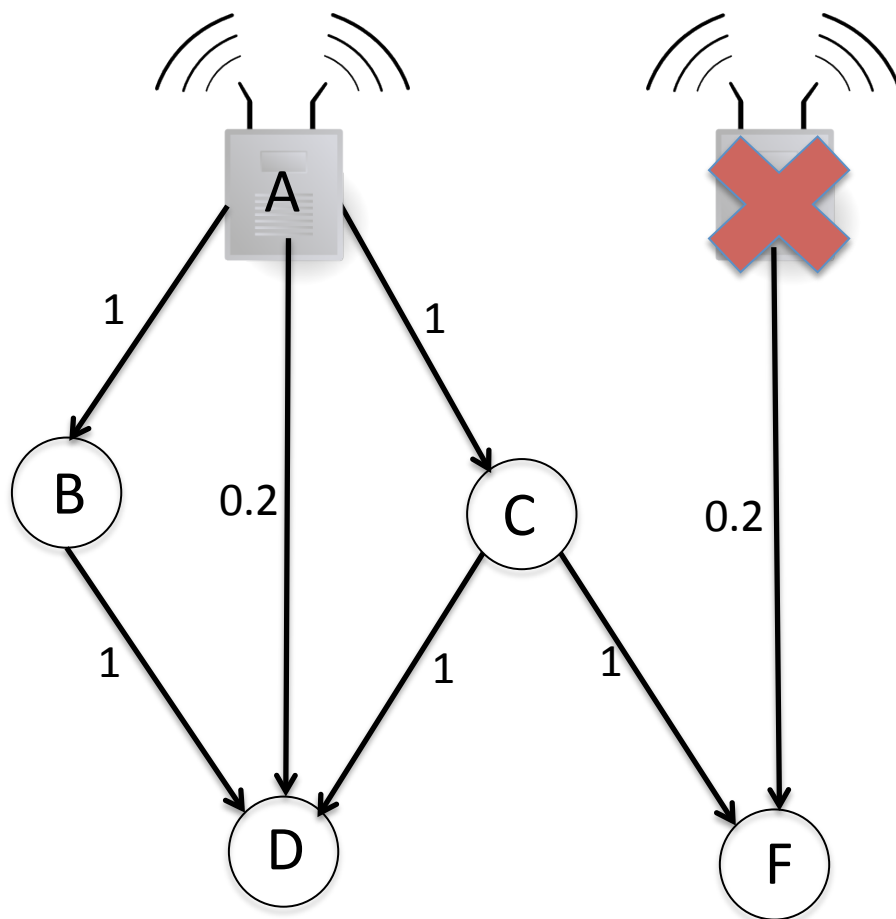
- Use efficient flooding to send data to all nodes subscribed to the multicast group

3. Bit-Rate Selection Mechanism

- Allow senders to choose the best bitrate



Why Client Cooperation?



Expected # Transmissions

with out client cooperation = 10
with client cooperation = 2

Benefits of Client Cooperation

1. Fewer transmissions → Improves multicast throughput
2. Lesser multicast traffic
3. Not all access points transmit

Challenges in Wireless Flooding

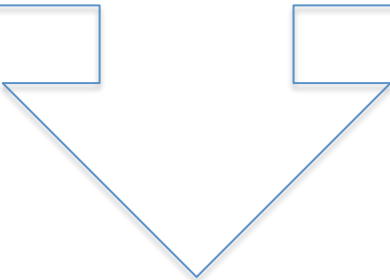
- Wireless receptions are probabilistic
 - How many packets to transmit?
- Pattern of packet reception is non-deterministic
 - What packets are with each receiver?
- Feedback is expensive
- Wireless transmissions are inherently broadcast
 - Two near by transmissions cannot coexist
 - How to exploit opportunism?

Design of UFLOOD

- Design questions
 - Who should transmit next?
 - What to transmit?
- UFlood's claim: Selection of best sender
 - Higher throughput
 - Fewer #transmissions

UFlood's Sender Selection Strategies

1. Favor higher delivery Probabilities
2. Favor senders with large number of receivers
3. Favor senders with new information
4. Account for correlated receptions



Utility = Value of a node's transmission
Best Sender ← Highest Utility

Computing Packet Utility

Utility(A)

Bit rate of node A

Indicates if transmissions of node A are useful to node B

$$U(A) = \sum_{B \in N_A} P_{A,B} \times b(A) \times I_{A,B}$$

Neighbors(A)

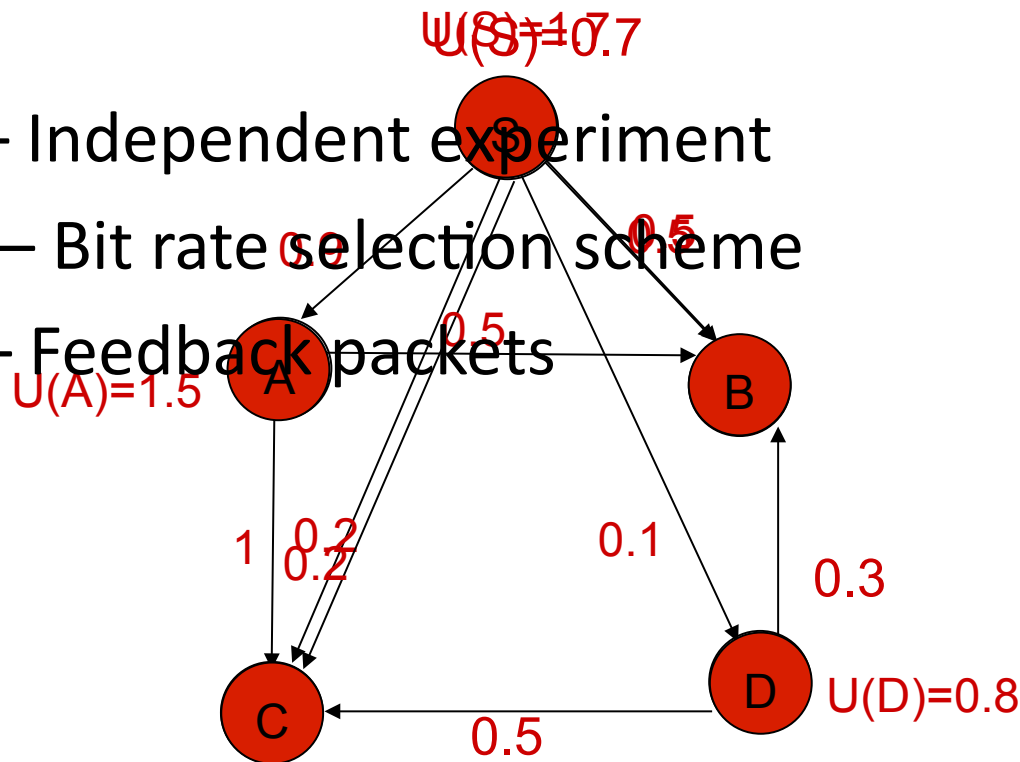
Delivery probability from node A to node B

The diagram illustrates the components of the packet utility formula. The formula is $U(A) = \sum_{B \in N_A} P_{A,B} \times b(A) \times I_{A,B}$. Arrows point from descriptive text to each part of the formula: 'Utility(A)' points to $U(A)$, 'Bit rate of node A' points to $b(A)$, 'Indicates if transmissions of node A are useful to node B' points to $I_{A,B}$, 'Neighbors(A)' points to N_A , and 'Delivery probability from node A to node B' points to $P_{A,B}$.

How UFLOOD works?

$$U(A) = \sum_{B \in N_A} P_{A,B} \times b(A) \times I_{A,B}$$

- $P_{A,B}$ – Independent experiment
- $b(A)$ – Bit rate selection scheme
- $I_{A,B}$ – Feedback packets



Pseudo Code of UFlood

Packet preparation:

1. All APs receive the file from multicast server.
2. Split file in to equal sized packets
3. Group in to batches of 64 packets.
4. Batches are flooded one at a time.

Pseudo Code for UFlood

Random Network Coding

4. Source AP has “native” packets (n_1, \dots, n_{64})
5. Source constructs “coded” packets = Linear Combination or $LC(n_1, \dots, n_{64})$

$$P_1 = c_1 \cdot n_1 + c_2 \cdot n_2 + \dots + c_{64} \cdot n_{64}$$

$$P_2 = c_1 \cdot n_1 + c_2 \cdot n_2 + \dots + c_{64} \cdot n_{64}$$

...

...

- These are first generation packets

Pseudo Code of UFlood

6. UFlood is distributed and a local heuristic: Nodes periodically calculate utility of itself and all its neighbors
7. The best sender transmits coded packets in burst.
8. All nodes recode every time a packet is sent
9. Nodes broadcast feedback of the packets they possess.
10. Go to Step 6.

Implementation Issues: Feedback

- $I(A,B)=1$ if transmissions of A are linearly independent to packets of B
- How to construct feedback for Coded packets?
 - Coefficients of each coded packet – Huge!
 - Rank = # Linearly independent coded packets
 - bitmap identifying each distinct first-generation packet that contributed via coding to any of the packets B holds
 - Feedback \leftarrow Rank(B)+bitmap+Rank(N(B))
- How often to send feedback?
 - Smart feedback
 - Nodes interpolates feedback
 - Detects an idle channel for 3-pkt duration

Implementation Issues: Deadlocks

1. Feedback packets includes neighbor's rank –
Two hop information → Accurate utility
calculation of neighbors
2. Sends burst of packets → Reduces
#deadlocks

Implementation Issues: Burst size

- Burst size = $\min_{B \in N(A)} (L_{A,B})$
- $L_{A,B}$ = # Packets A can send to B without causing utility(B) to be greater than utility(A)

Contributions of UFlood

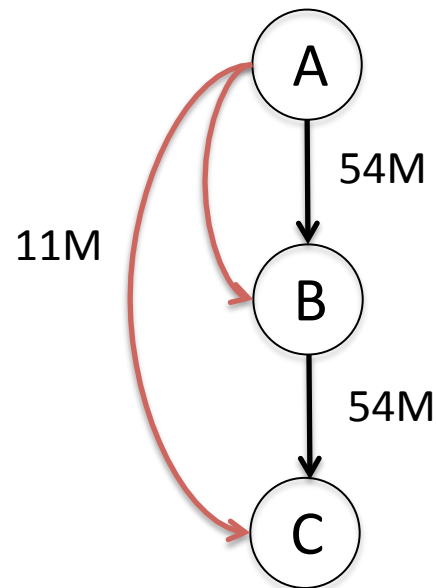
- Notion of *Utility* – Sender selection
- Smart feedback for coded packets
- A distributed implementation

Lower Bit Rates are Slow but Strong

- $P_{A,B}$ at $b1 \leq P_{A,B}$ at $b2$, if $b1 > b2$
- $P_{A,B}$ at 1Mbps = 1, then $P_{A,B}$ at 54Mbps ≤ 1

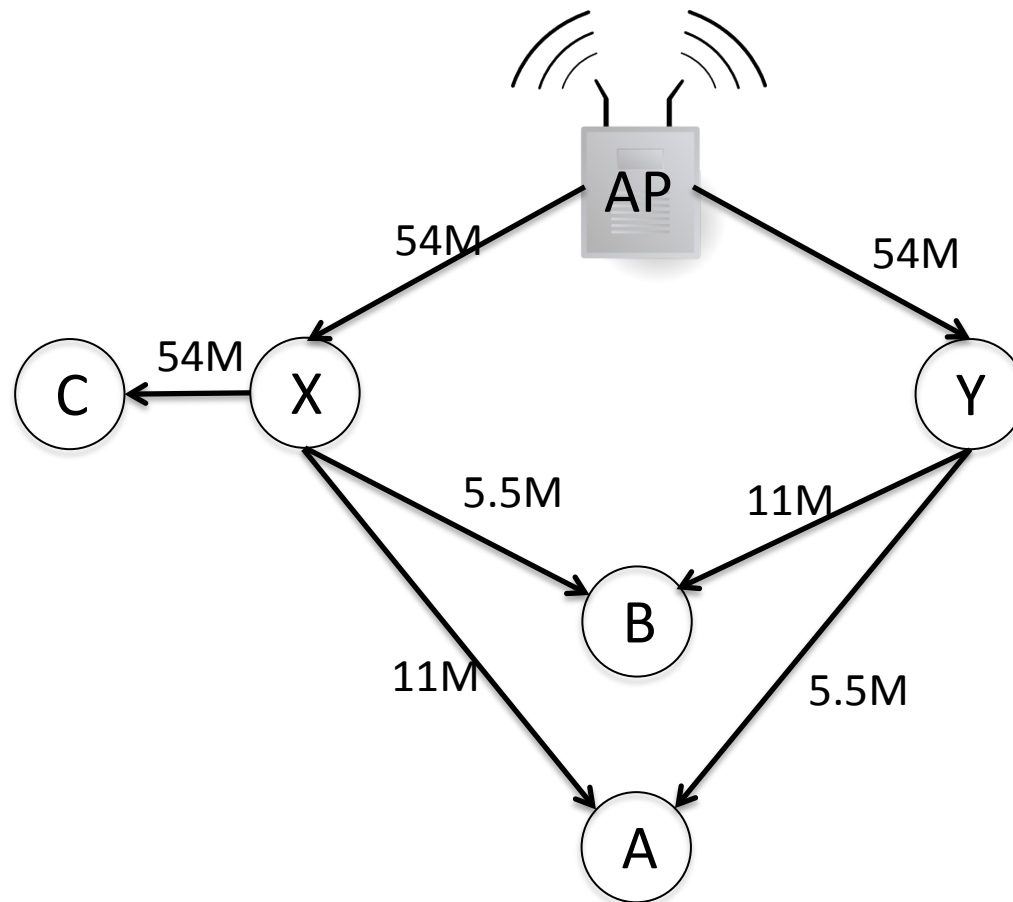
Challenges in Bit Rate Selection

- Single hop (Lower rate) Vs Multi-hop (Higher rate)



Challenges in Bit Rate Selection

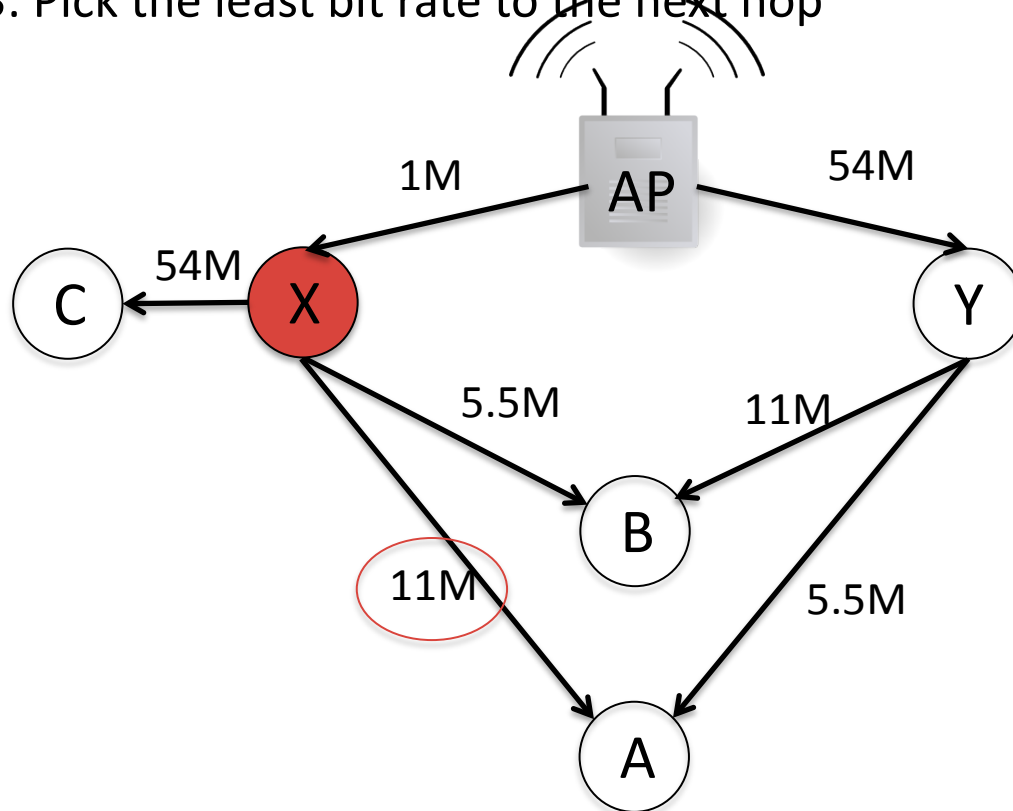
- Many senders and Many Receivers



Net Rate = 15M

Bit Rate Selection for Node X

- Step 1: $ETT(X,C,b) = 1/(P_{X,C} * b)$
- Step 2: Best bit rate for link XC = $\min_b ETT(X,C,b)$
- Step 2: Construct Dijkstra shortest path routes from AP to all the nodes, using ETT metric
- Step 3: Pick the least bit rate to the next hop



Implementation

- 6 APs and 20 nodes on a 250x150meters 3-floor office building
- Nodes: 500 MHz AMD Geode LX800 CPU
- 802.11b/g, Omni-directional antenna
- Transmit power = 12 mW
- CLICK software router toolkit
- Carrier Sense on

Performance Comparison

Metrics:

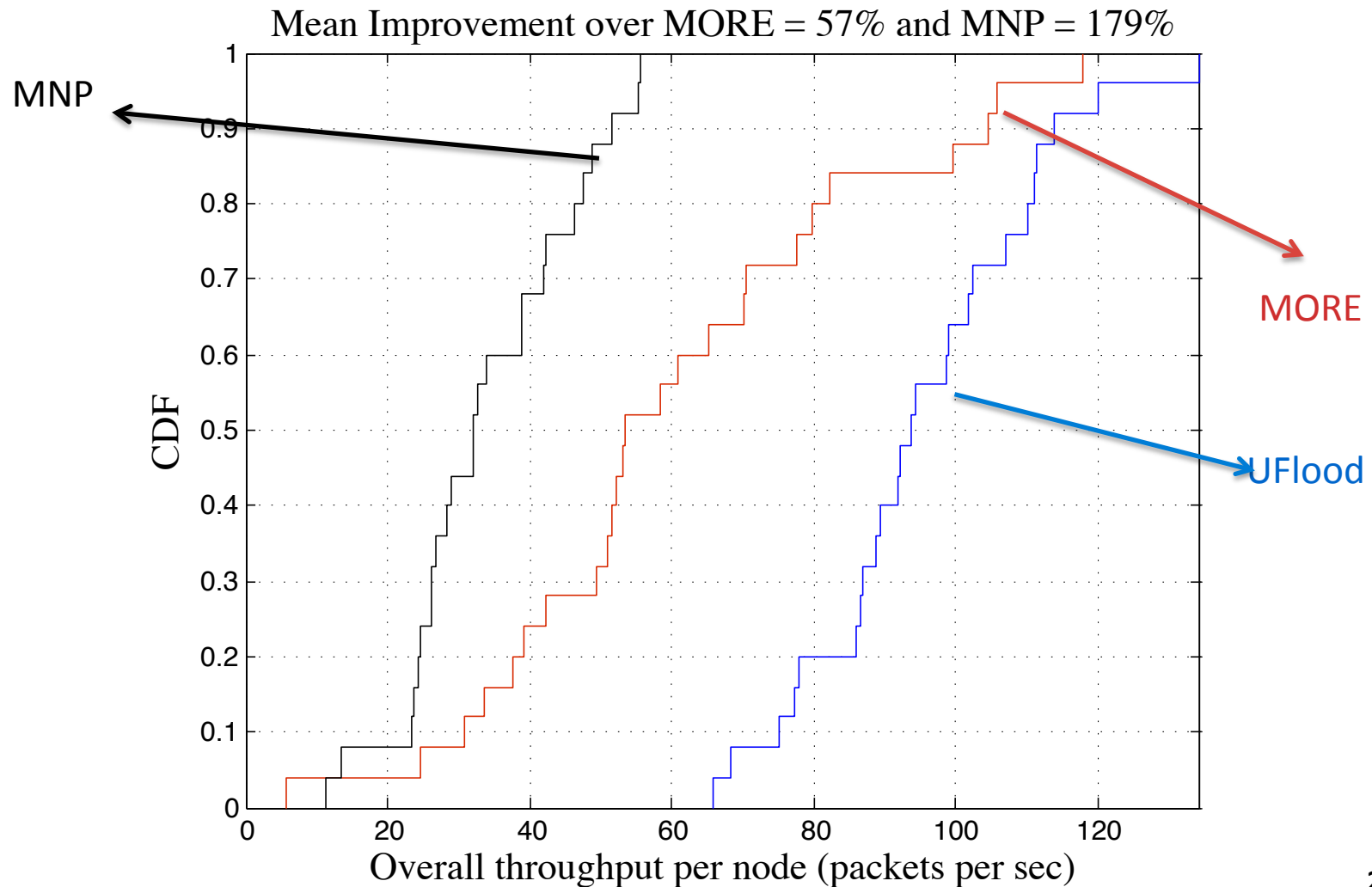
$$\text{Throughput}(PPS) = \frac{\text{TransferSize}}{\text{Packet size} \times \text{Total time to complete flooding}}$$

$$\text{Airtime}(Sec) = \sum_{i=1}^N \text{Time spent by node } i \text{ in transmitting packets}$$

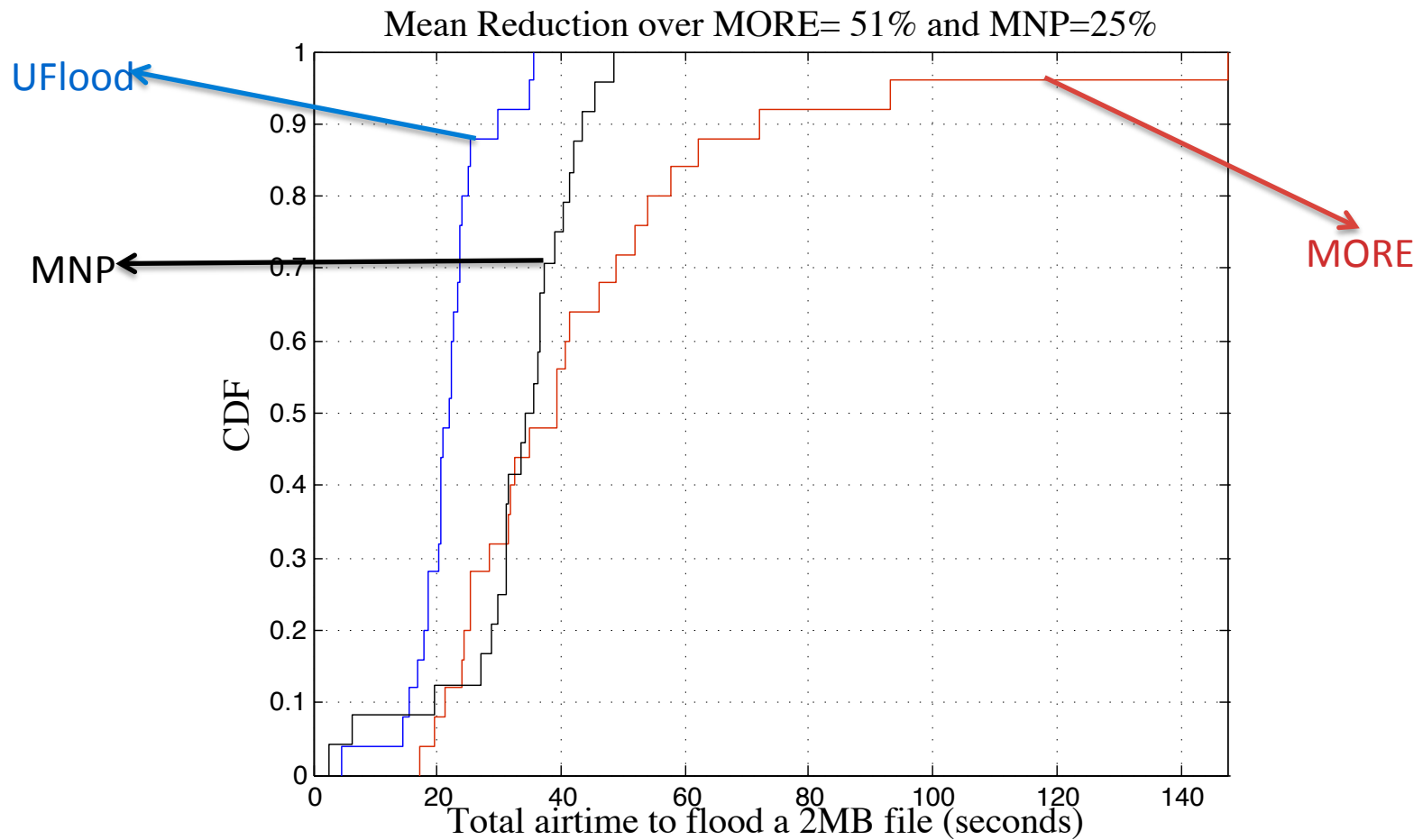
Protocols used for Comparison

- UFlood Vs MORE
 - Statically assigns the number of packets a node sends for each packet reception
 - No detailed feedback
 - High throughput but wasted transmissions
- UFlood Vs MNP
 - Save Energy
 - Too slow but efficient transmissions

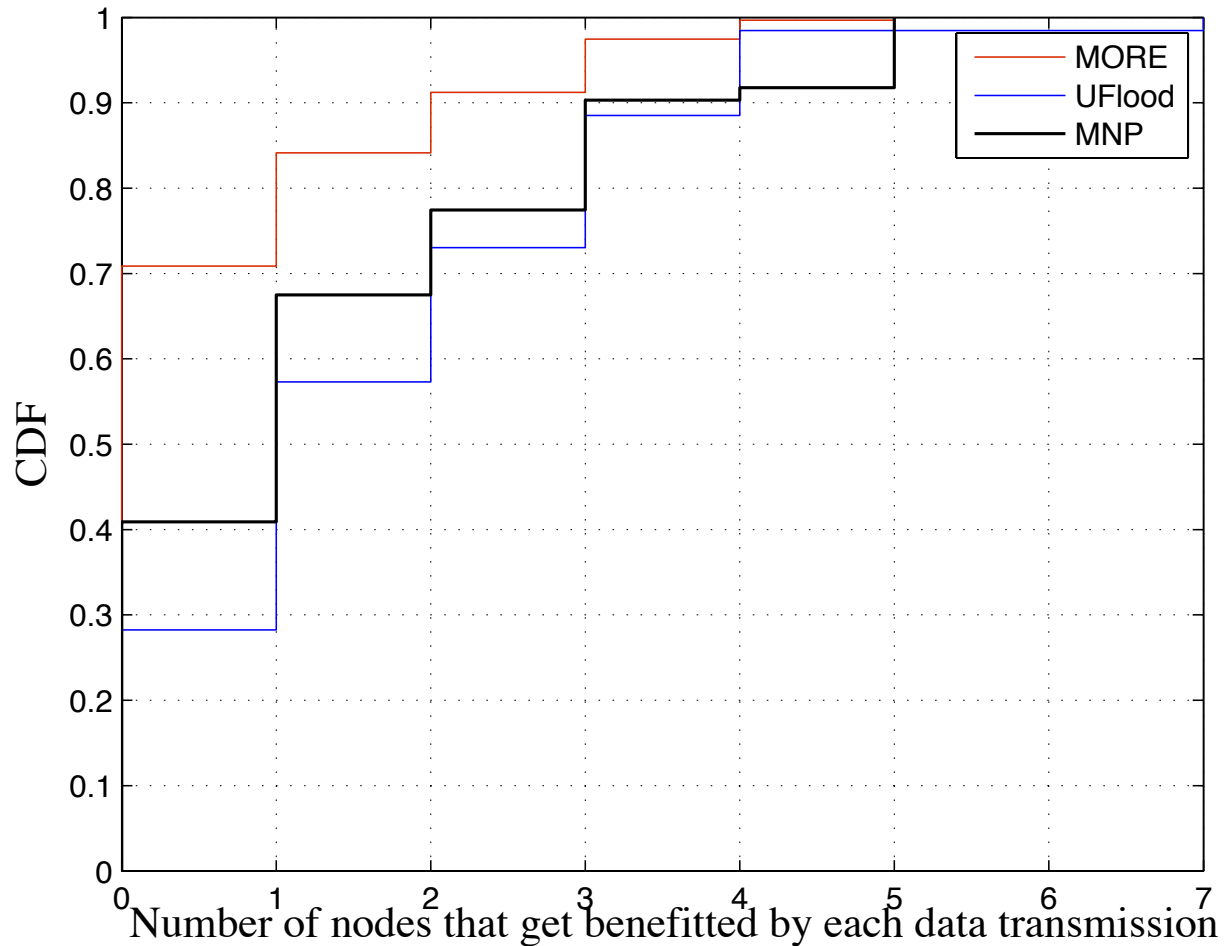
UFlood: Throughput



UFlood: Airtime



Why UFlood Wins?



Each UFLOOD transmission benefit twice as many receivers as MORE and 20% more than MNP

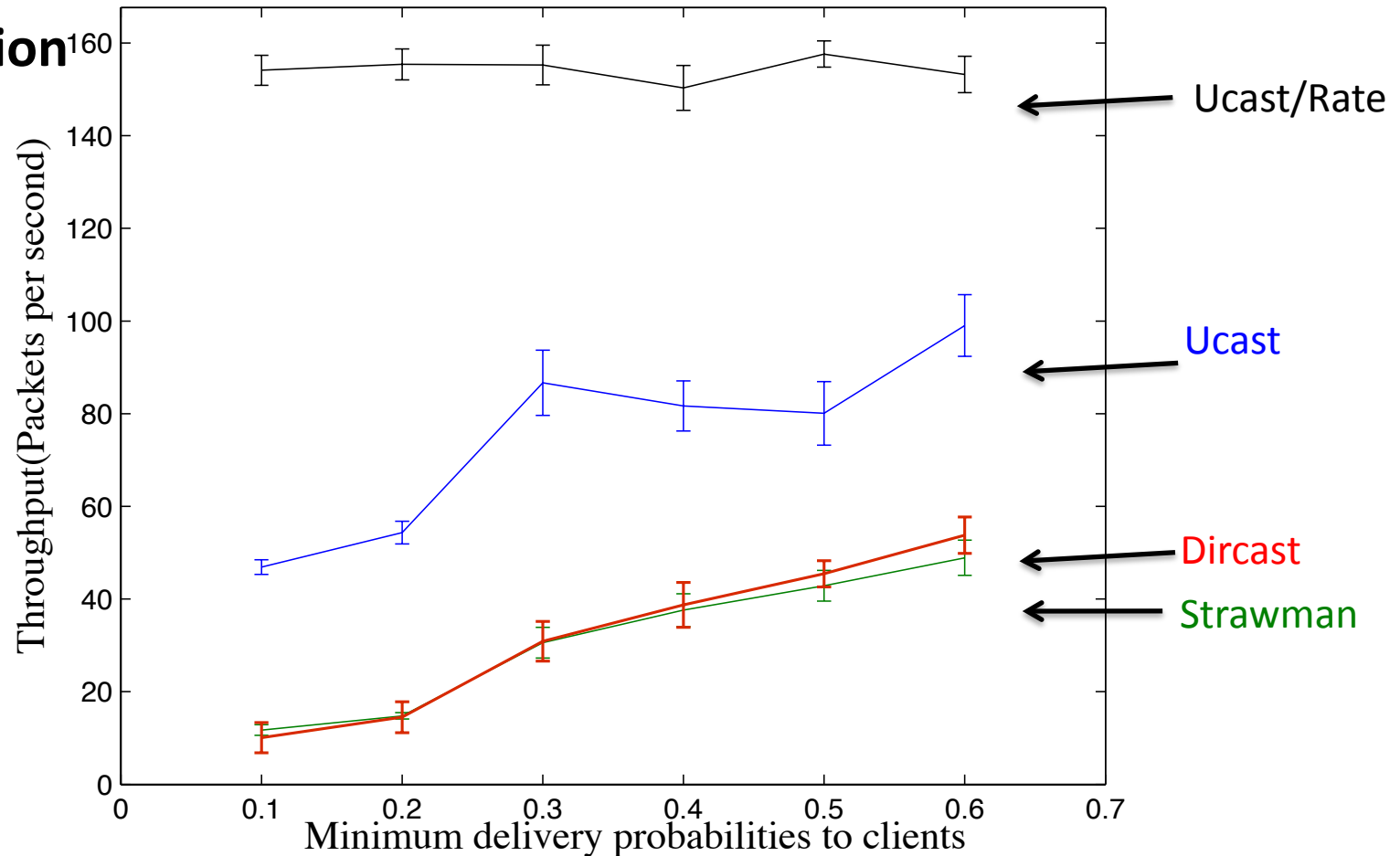
Protocols used for Comparison

- UCast
 - Constant Bitrate of 5.5Mbps
- Ucast/Rate
 - Use Bit rate selections
- Strawman
 - Traditional WiFi multicasting
 - N/w coding
- Dircast
 - AP sends packets until the poorest receiver receives all the packets
 - N/W coding
 - Rate selection for APs

UCast Vs Dircast VS Strawman:

Throughput

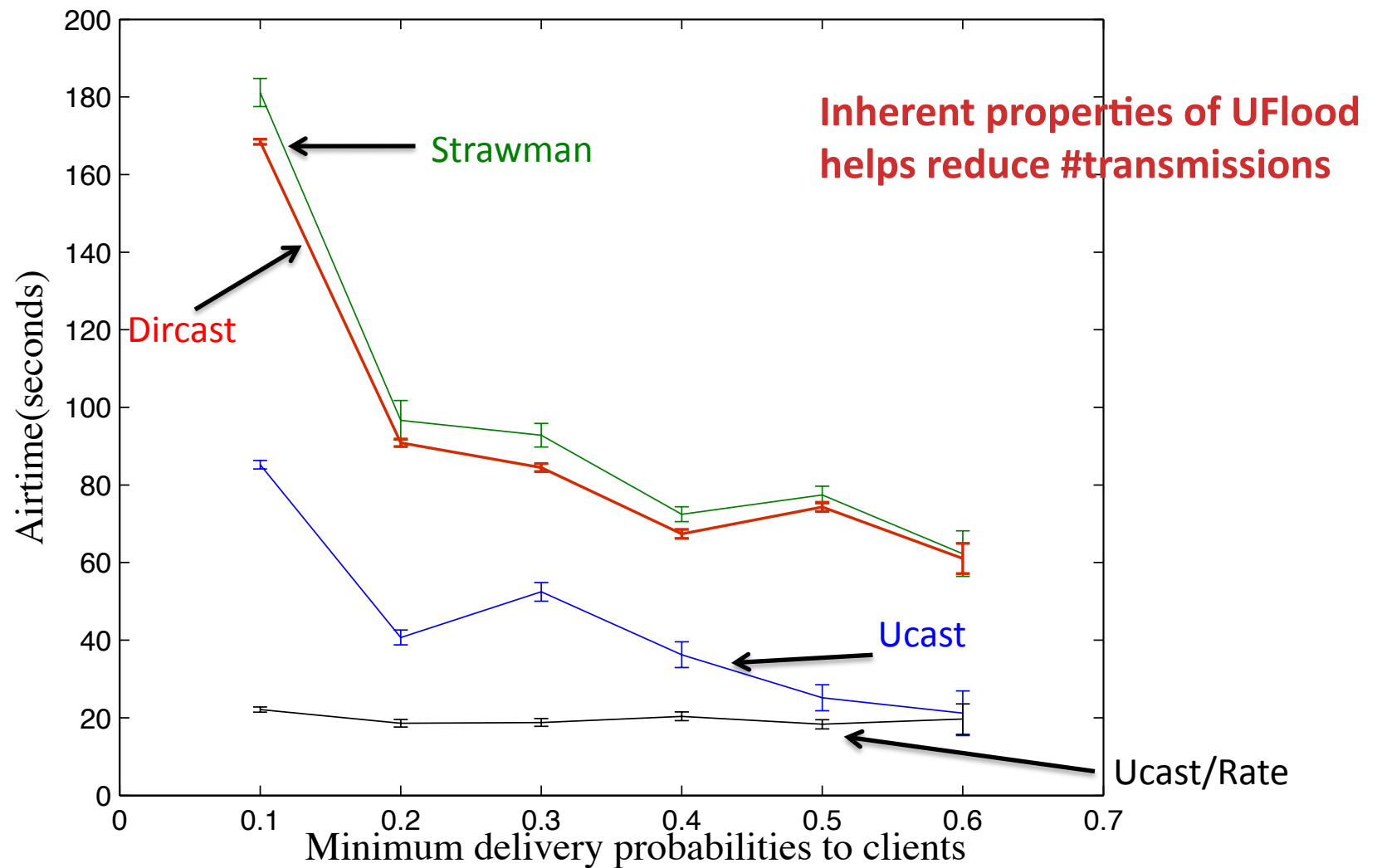
Insensitive to AP
connection



Client coopertion

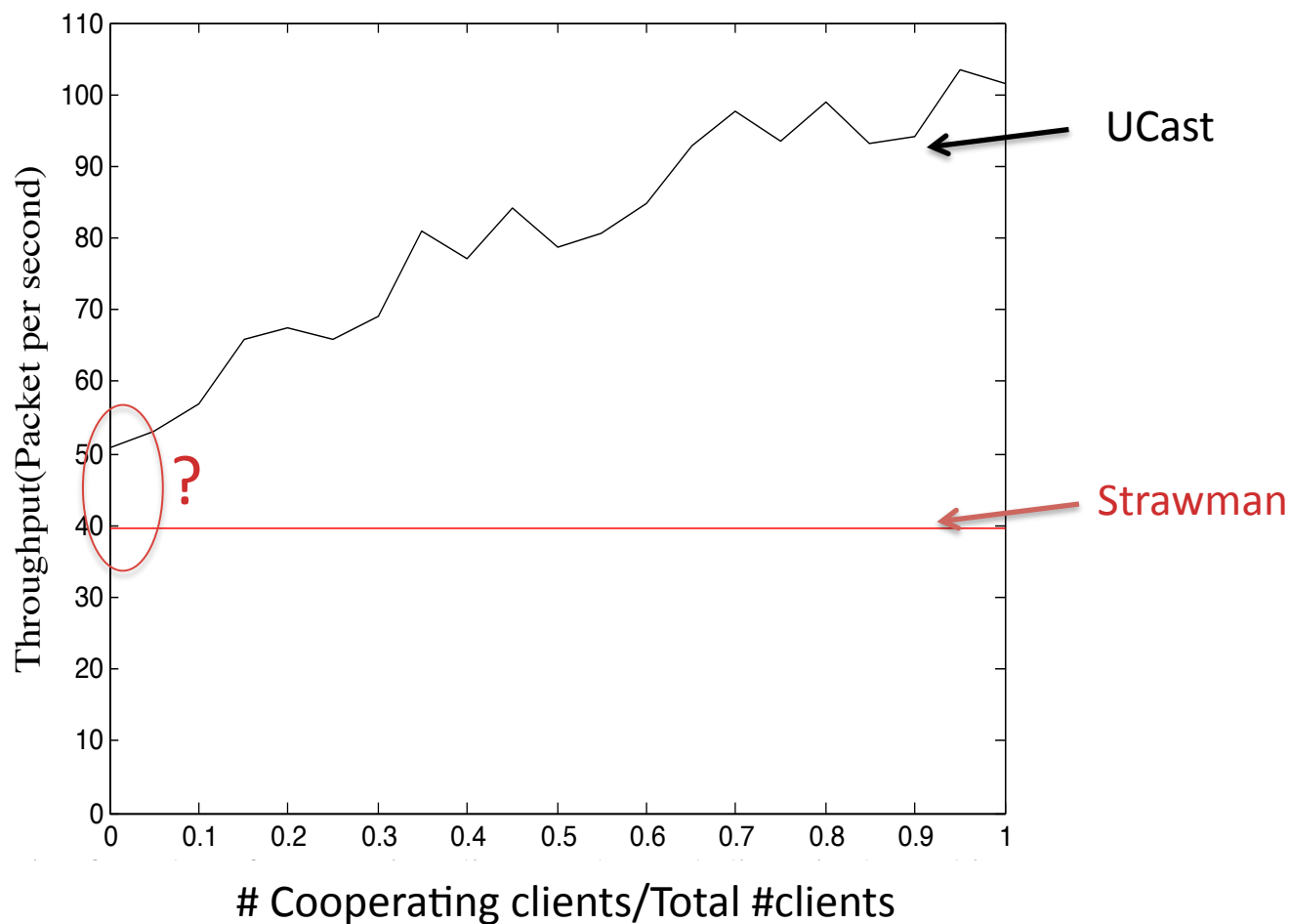
Few APs never send!

UCast Vs Dircast VS Strawman: Airtime



Why Client Cooperation?

**Only best
APs send**



Contributions of this work

- UCast: Client cooperation multicasting and experiments show a huge benefit
- UFlood: High-throughput distributed flooding scheme
 - Introduce notion of **Utility**
 - **Smart feedback** for coded packets
 - Increases throughput and uses fewer transmissions
- A novel bit rate selection for flooding protocols