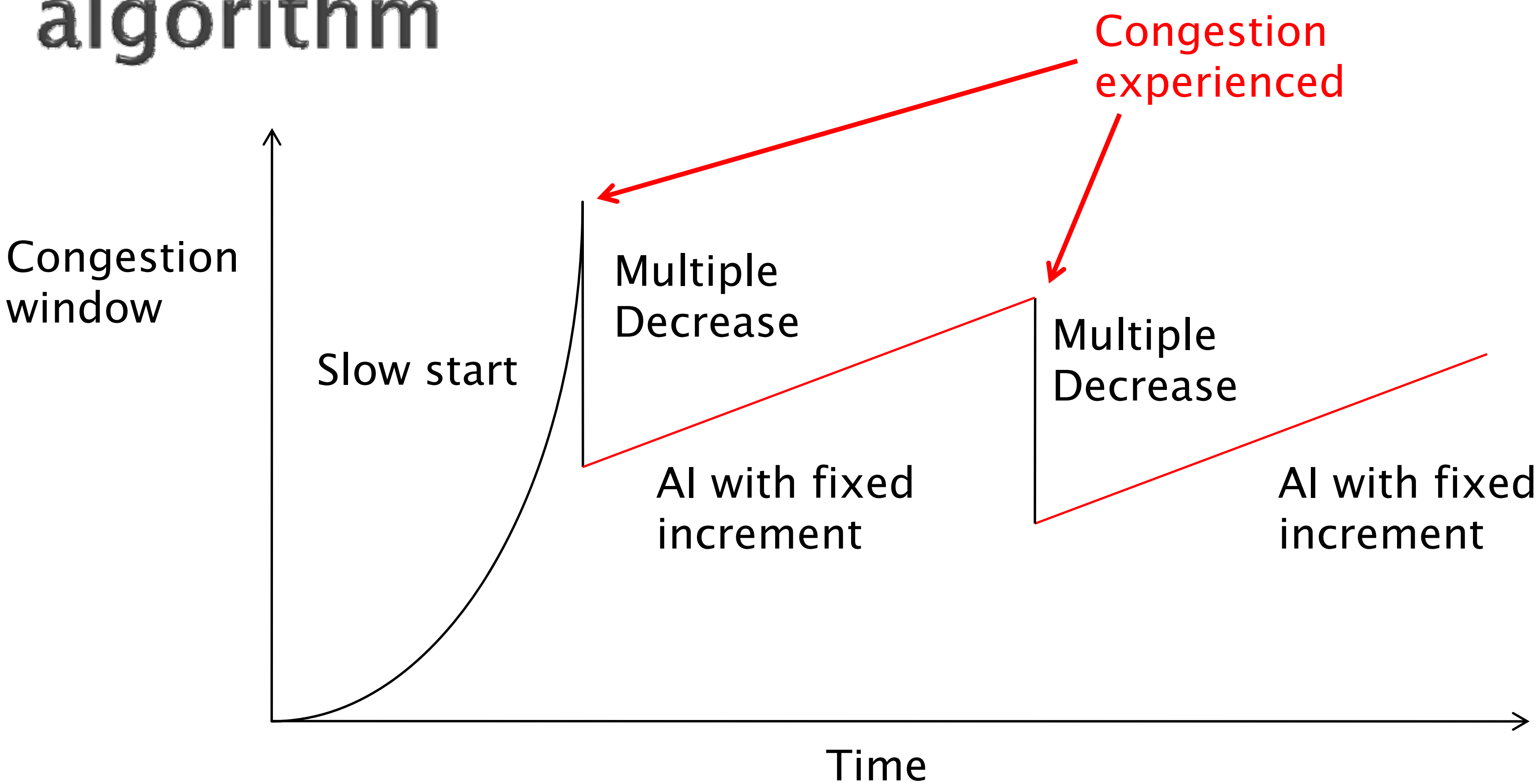
Uno – A framework for deploying explicit feedbak congestion control protocols

Nedeljko Vasić (EPFL, Switzerland)
Srinidhi Kuntimaddi (IIT Guwahati)
Dejan Kostić (EPFL, Switzerland)

Motivation

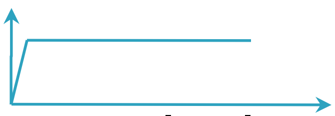
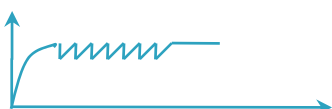
- Bandwidth-delay product of Internet links keep increasing
- Application such as VoIP, VoD are putting even more stress on the Internet links
- Issues with Congestion Control (CC) algorithms such as TCP:
  - Slow convergence
  - Lack of fairness among flows with different RTT

Slow convergence of TCP's AIMD algorithm



Additive Increase takes a lot of time to reach up to maximum value

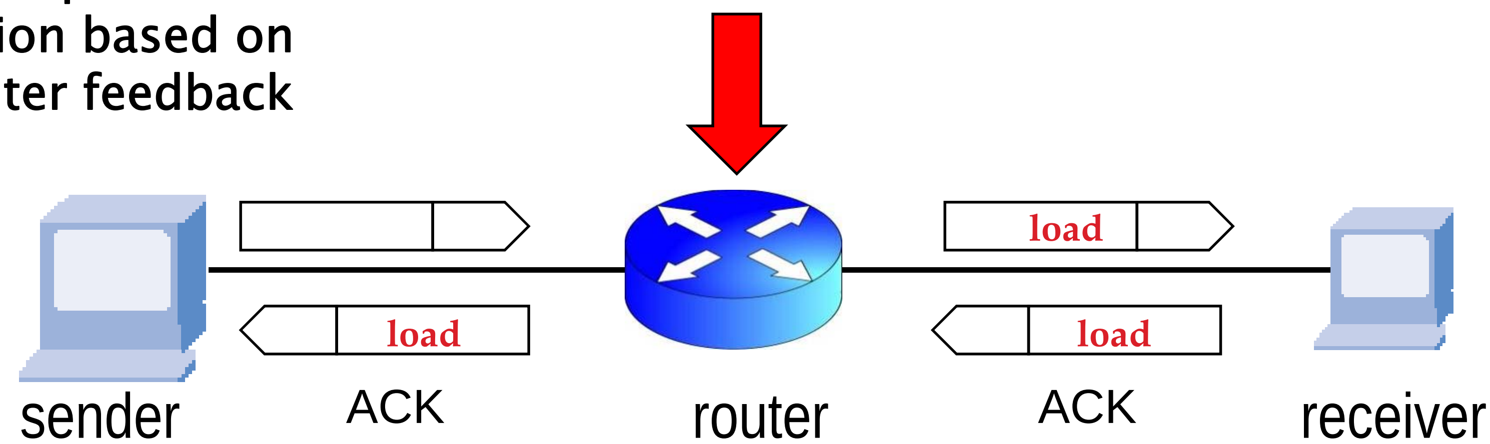
Existing Solutions

- End to end scheme with implicit feedback
  - TCP with all mentioned limitations 
- Network based scheme
  - XCP 
    - Routers calculate and distribute spare bandwidth fairly
    - Very efficient, but it needs 128-bit congestion header
- End to end scheme with explicit feedback
  - VCP, MLCP 

Load-based CC algorithms

Load encoded into the packets by the router using an encoding scheme

Sender takes appropriate action based on router feedback



VCP [Xia05]

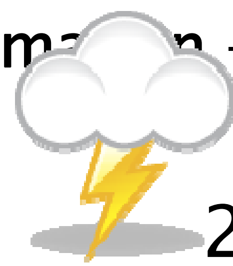
- Comes close to the ideal of:
  - 100% link utilization, no queuing
- Uses 2-bit feedback which can fit into existing 2 ECN bits in the IP header
  - Immediately deployable
  - Relatively long convergence times
- Does not propagate per flow RTTs
  - Causes fairness issues for flows that have very different RTTs

MLCP [Qazi08]

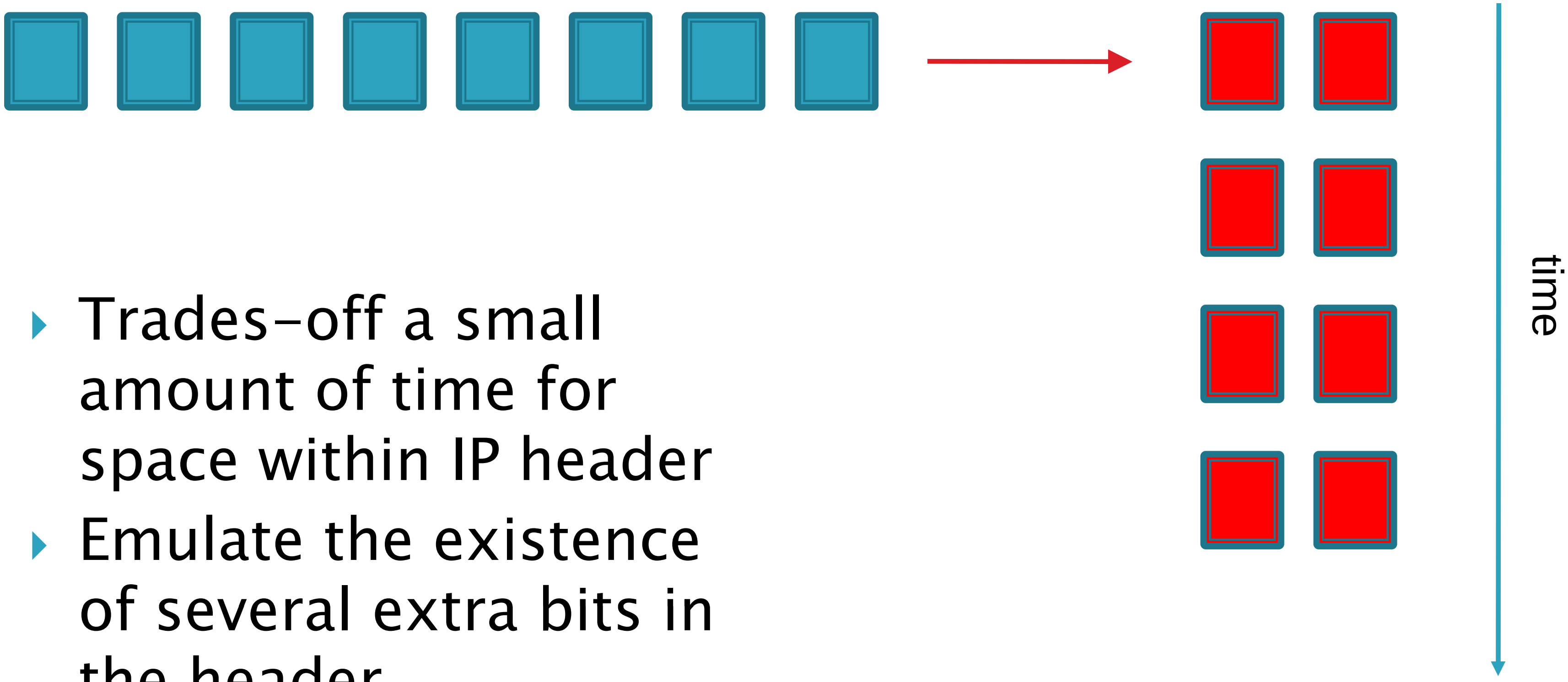
- Alleviates most of VCP's performance issues by:
  - Encoding more load levels
  - Transmitting the per-flow RTTs,
- Major findings: Sufficient to have
  - **Finer load-factor granularity** – 4 bits
  - **Round trip time inform[illegible]n** – 3 Bits

4 bits

3 more bits

- 7-bit information  2-bit space

Uno



- Trades-off a small amount of time for space within IP header
- Emulate the existence of several extra bits in the header

Uno

- Framework that allows the load-factor based congestion control schemes immediately deployable which
  - Uses existing fields in the IP header
  - Efficiently conveys the router load and/or RTT information to the sender

IP Header

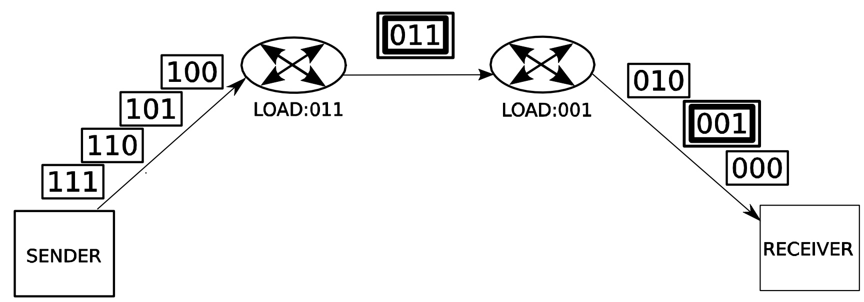
<table>
<tr><td>Version</td><td>IHL</td><td>TOS/DSCP/ECN</td><td colspan="3">Total Length</td></tr>
<tr><td colspan="3">Identification</td><td>Flags</td><td colspan="2">Fragment Offset</td></tr>
<tr><td colspan="2">Time To Live</td><td>Protocol</td><td colspan="3">Header Checksum</td></tr>
<tr><td colspan="6">Source Address</td></tr>
<tr><td colspan="6">Destination Address</td></tr>
<tr><td colspan="5">Options</td><td>Padding</td></tr>
</table>

Outline

- UNO framework
  - Load factor information
  - RTT related information
- Implementation
- Evaluation
- Conclusion & Future Work

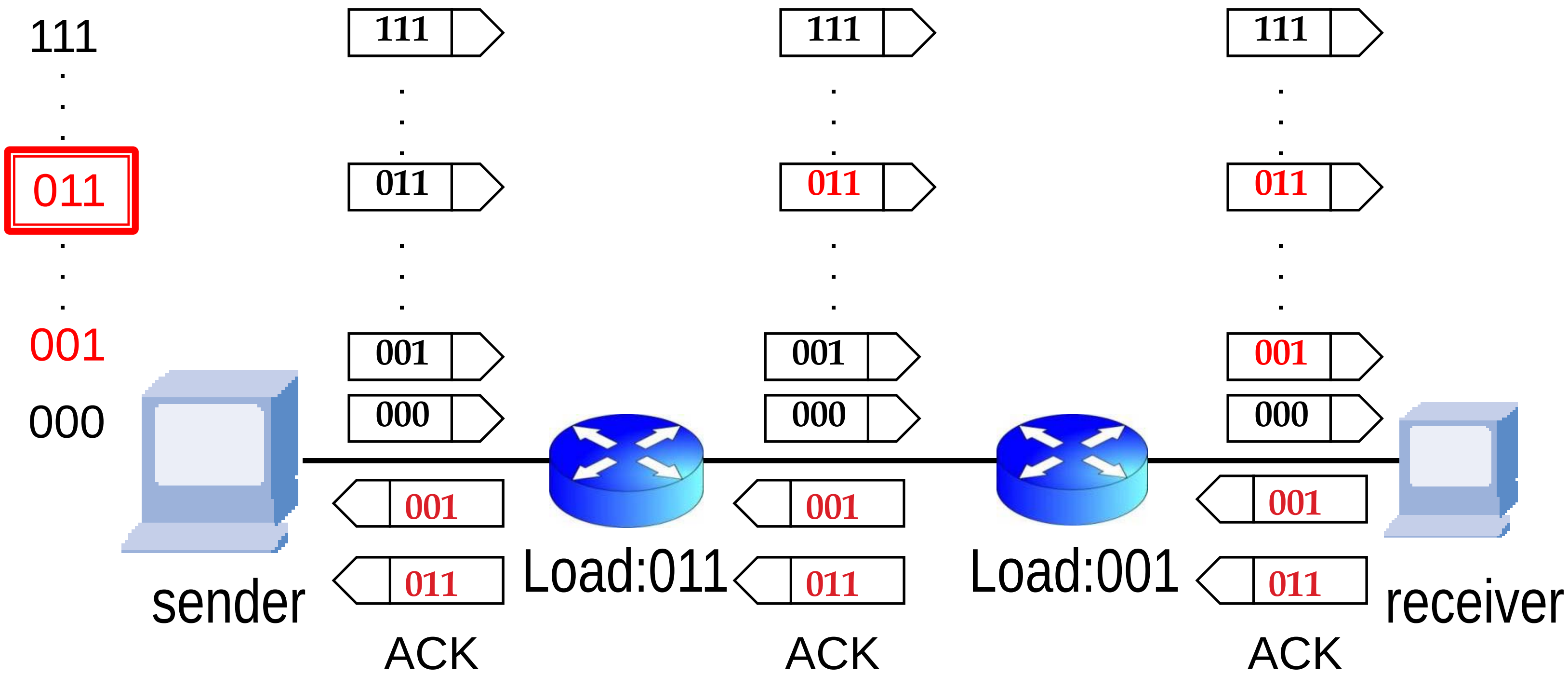
UNO – transmitting link load

- Uses ECN bits already present in the IP header
- Makes use of the IP Identification number (IPID) field to mark packets
  - Uniformly increasing IPIDs
  - Random IPIDs
- Provides way for router to encode its load information into the packet without the use of more fields



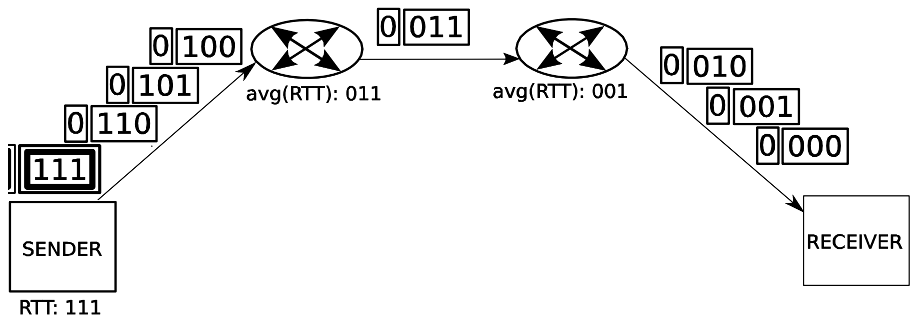
Using 1 UNO bit to transmit link load information to the receiver

UNO – transmitting link load



UNO – transmitting RTT

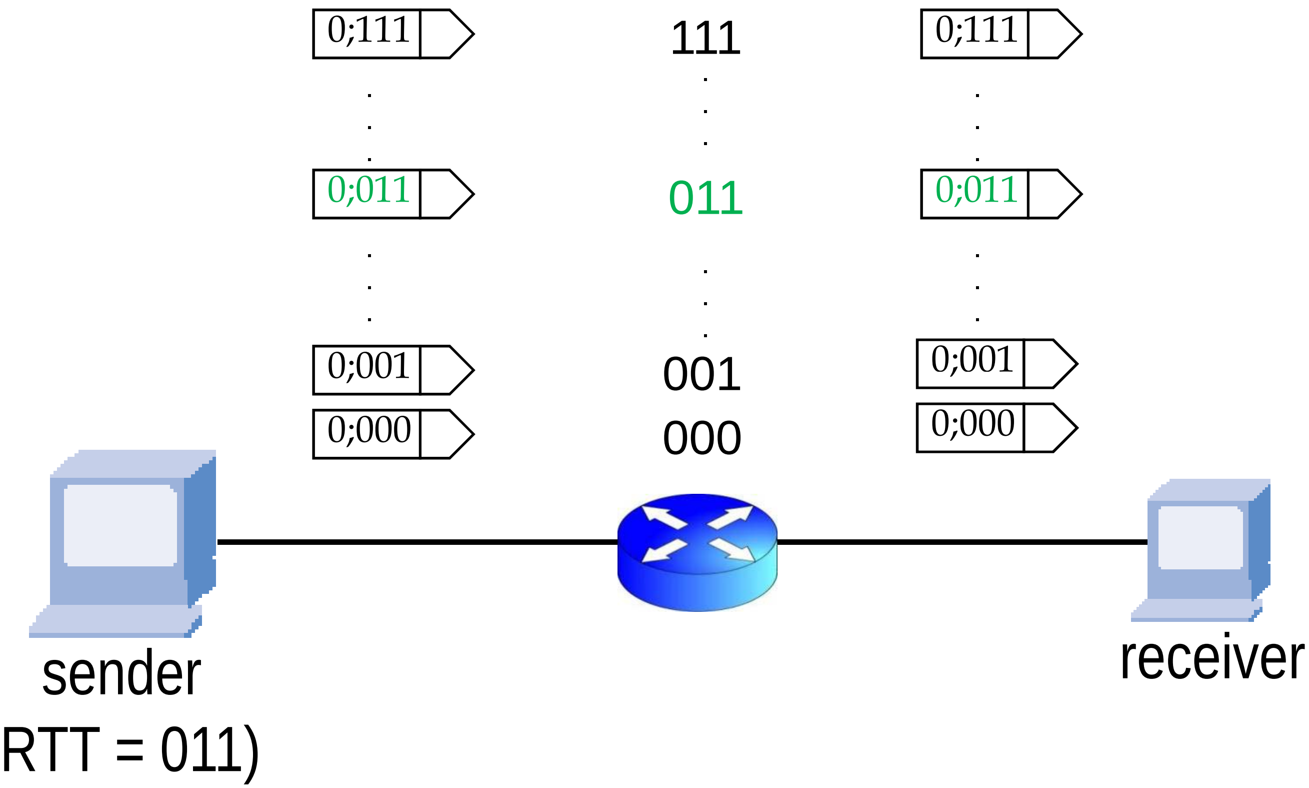
- Two phases
- Uses (N+1)st right-most bit to tell the phase:
  - Sender->Router
  - Router->Receiver



Transmitting per-flow RTT to the routers using the second UNO bit

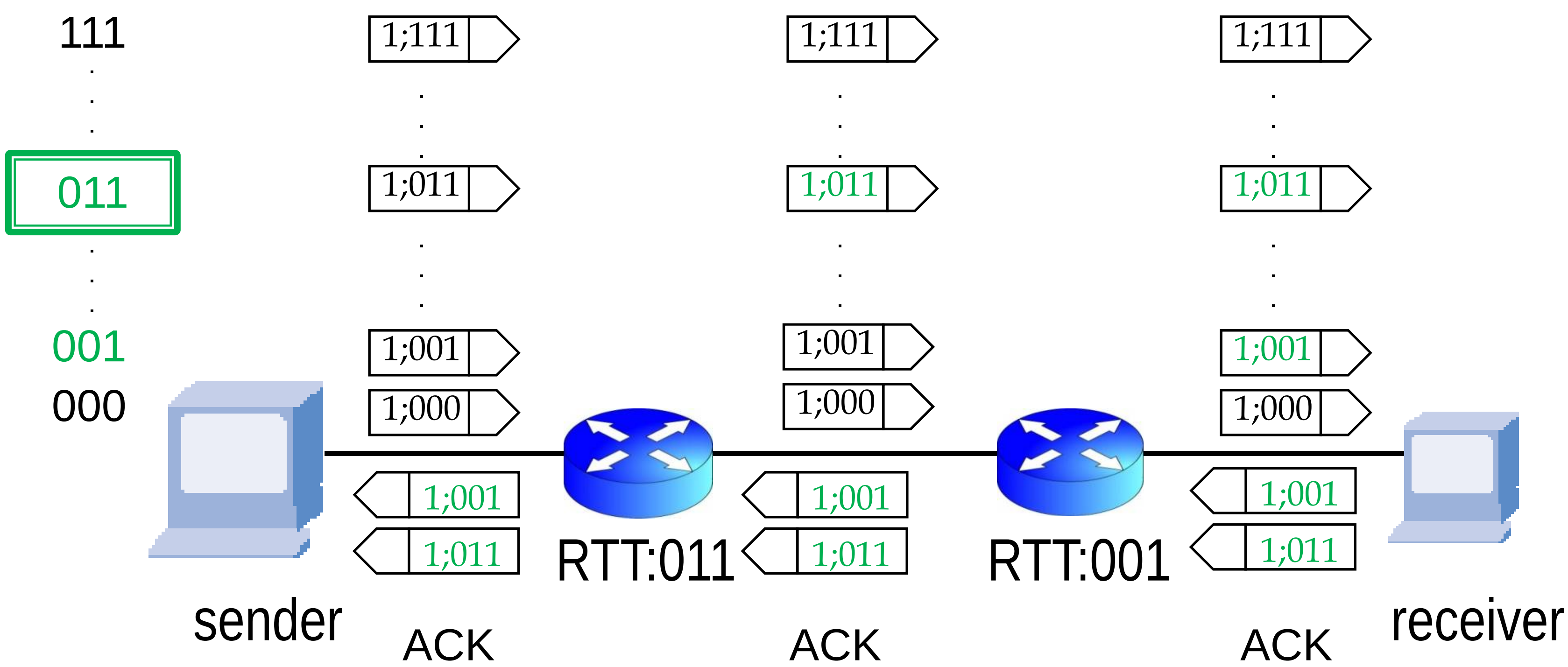
UNO – transmitting 3-bit RTT

Sender->Router



UNO – transmitting 3-bit RTT

Router->receiver



UNO-discussion

- Has to wait for 2^N packets for monotonically increasing IPIDs
- Only a small delay of 0.07 ms on a 1 Gbps link with 1000 byte packets, using 3 bit load encoding
- Random IPIDs – instance of coupon collector problem

UCP1 Implementation

- VCP protocol implemented over UNO framework (called UCP1)
- Using 3 bits to encode load information as opposed to two bits in VCP
- Compared against VCP implementation with 3 bits (called VCP3)

UCP2 implementation

- UCP2 uses RTT information for congestion control
- Uses MLCP's RTT encoding scheme – Quantized average RTT takes a value from
  - $S = \{80,200,400,600,800,1000,1200,1400\}$

Evaluation

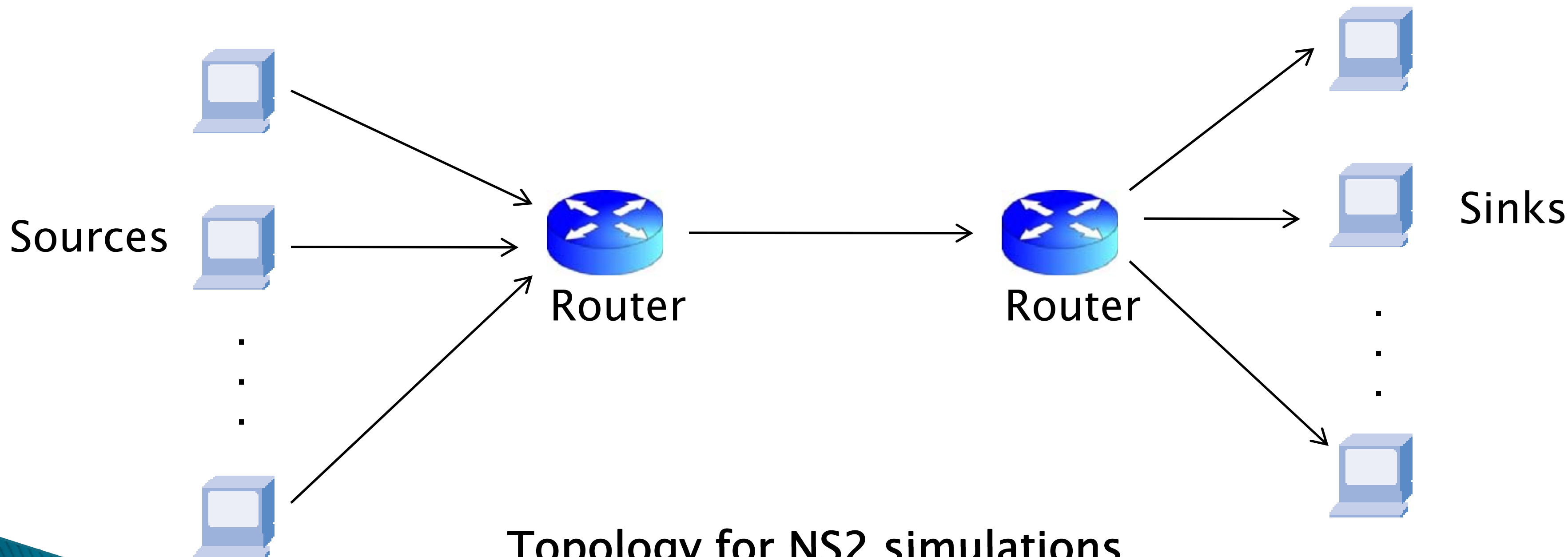
1) Does UNO framework help in quickly obtaining high link utilization?

2) Can UNO framework enforce fairness?

3) What is the impact of additional latency caused by waiting for 2 power of N packets?

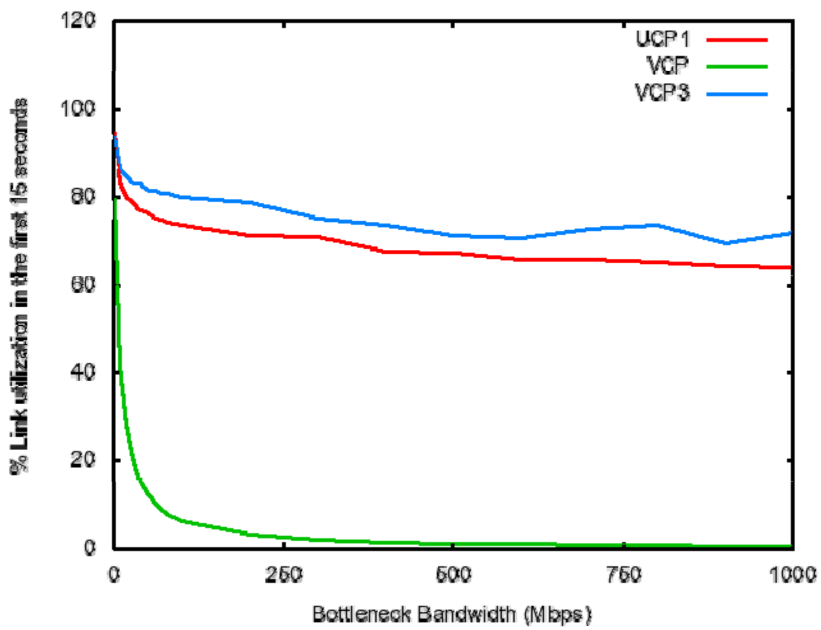
Evaluation

- NS2 simulation under various network topologies
- Varying link capacities and latencies



Topology for NS2 simulations

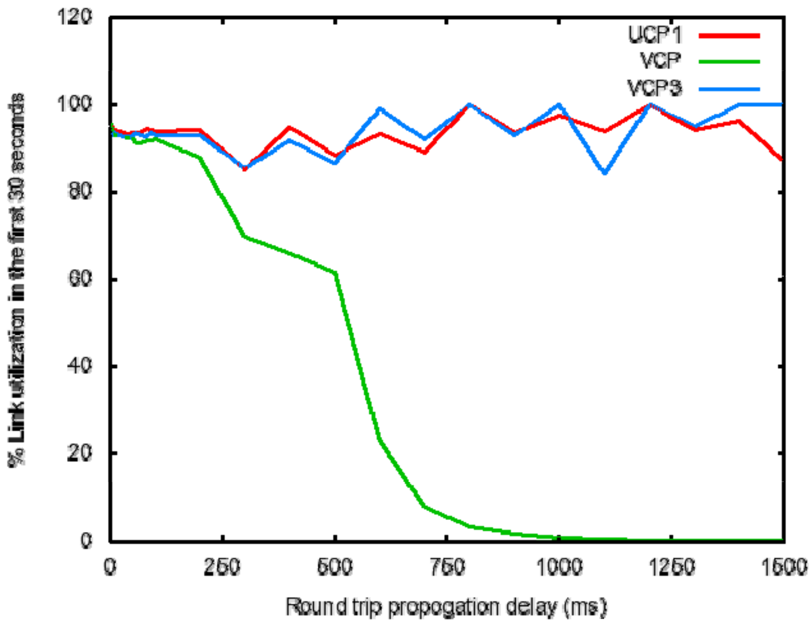
Varying bandwidth (MI phase)



- UCP1 performs better than VCP almost as good as VCP3
- After first 15 sec

UCP1 converges quickly

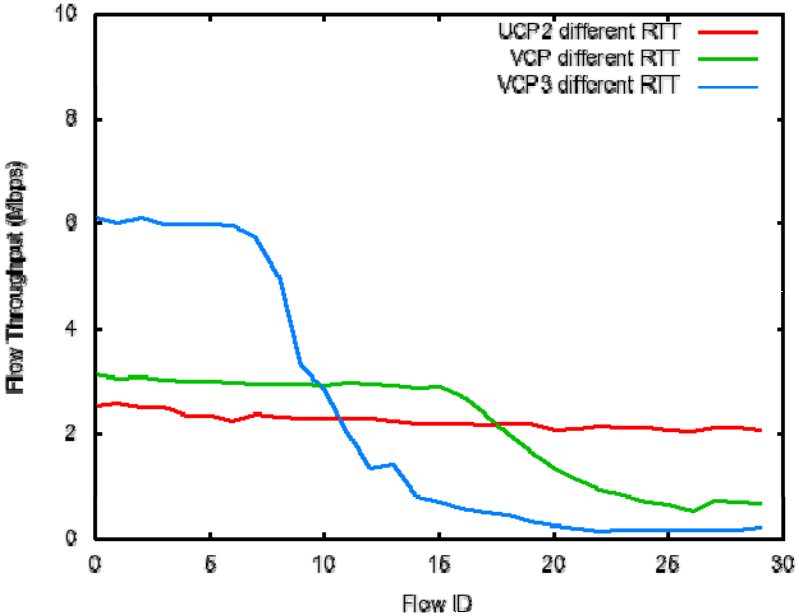
Varying RTT (MI phase)



- fix the bottleneck capacity to 45Mbps and vary the round-trip time
- After 30 sec

UCP1 outperforms VCP

Fairness for flows with different rtts



Transmitting average RTT improves fairness

- 30 FTP flows with RTTs ranging from 40ms to 156ms

Conclusion

- Single ECN bit is enough for load-factor based congestion control schemes
- Schemes using load and RTT are immediately deployable without changes made to the IP header
- 1-bit scheme interoperable with existing ECN implementation

Future work

- Implementation of other load factor based congestion control protocols
- Checking performance in the presence of non-UNO, ECN routers

Thank you

Related work

Other schemes that make use of IPID field

- Probabilistic Packet Marking (PPM) [Adler02]
- Deterministic Packet Marking for congestion price estimation [Thommes06]
- Adaptive Deterministic Packet Marking (ADPM) [Andrew06]
- Deterministic packet marking for Max-Min flow control [Ryu05]
- Exploiting the IPID field to infer network path and end-system characteristics [Chen09]

Our Scheme

- Implementation and evaluation of load-factor based protocols
- Transmit RTT information along with load