Problems with TCP Reno

- AIMD (Additive Increase, Multiplicative Decrease) algorithm of TCP Reno in congestion avoidance phase:
  - No packet loss (AI): increase congestion window by one packet/RTT too slowly
  - Packet loss (MD): decrease congestion window by half too dramatically
- Example: For fully utilize a link of 10 Gbps with
  - Packet size: 1,500 bytes
  - RTT (Round Trip Time): 100 ms
This requires:
  - Congestion window = 83,333 packets ➔ Packet drop rate = 2 \cdot 10^{-4}
It is impossible with present optical fiber and router technology.

HighSpeed TCP – HSTCP

- AIMD of HSTCP
  - AI: 1 \cdot a(w)
  - MD: 0.5 \cdot b(w)
- Metrics: Throughput and fairness against TCP Reno
- Simulation
- TailDrop, different access link bandwidth.

Objectives

- Consider the mixture situation of TCP Reno and HSTCP
- Evaluate the performance of HighSpeed TCP
  - Simulation
  - Metrics: Throughput and fairness against TCP Reno connections
- Propose ‘Gentle’ HighSpeed TCP
  - Higher throughput and better fairness than the original HighSpeed TCP

Evaluation

Case 1: TCP Reno is used for S₁ and HSTCP+3ACK is used for S₂

- Problems with HSTCP
  - Under-utilization (Case 2)
  - Unfairness (Case 3)
- Reasons
  - Buffer overflow leads that many packets are lost in bursty fashion
  - HSTCP still opens its CWND quickly even when packets are going to be stored in the router buffer
### Gentle HighSpeed TCP – gHSTCP
- 2 modes in congestion avoidance phase
  - Observes the packet transmission time and its RTT.
  - If the positive correlation is recognized, use gHSTCP Mode.
  - Otherwise, use HSTCP Mode.
- gHSTCP:
  - Prevent bursty packet drop.
  - Provide suitable fairness to the competing TCP Reno flows.

### Why introduce ARED
- Problems with TailDrop
  - Burst packet drop.
  - Global synchronization.
  - Unfairness.
- RED (Random Early Detection) is recommended.
  - But control parameters in RED is highly sensitive to the network condition.
- ARED (Adaptive RED)
  - An improved version of RED.

### ARED – Adaptive RED
- Difference between RED and ARED
  - Key point of ARED:
    - \( \text{max}_p \) is dynamically adapted to keep the average queue size within the target queue boundaries according to network conditions.
- Target queue range:
  \[
  [\text{min}_\text{th} + 0.4(\text{max}_\text{th} - \text{min}_\text{th}), \text{min}_\text{th} + 0.6(\text{max}_\text{th} - \text{min}_\text{th})]
  \]

### Algorithm used by ARED
- Sketch of ARED
  - If \( \text{avg} > \text{target} \) and \( \text{max}_p < \text{top} \):
    - Increase \( \text{max}_p \).
  - If \( \text{avg} < \text{target} \) and \( \text{max}_p > \text{bottom} \):
    - Decrease \( \text{max}_p \).
Gentle Adaptive RED -- gARED

Sketch of gARED:

- If \( \text{avg} > \text{target} \) and \( \text{avg} > \text{old}_{\text{avg}} \) and \( \text{max}_{\text{p}} < \text{top} \), increase \( \text{max}_{\text{p}} \).
- If \( \text{avg} < \text{target} \) and \( \text{avg} < \text{old}_{\text{avg}} \) and \( \text{min}_{\text{th}} < \text{avg} \), decrease \( \text{max}_{\text{p}} \).

Evaluation

Case 9: TCP Reno is used for \( S_1 \), and gHSTCP+SACK is used for \( S_2 \).

2 Cases: Case 8, Case 9.

TCP Reno Flows

TCP Reno Flows

Case 8: TCP Reno is used for \( S_1 \), and gHSTCP is used for \( S_2 \).

Case 9: TCP Reno is used for \( S_1 \), and gHSTCP+SACK is used for \( S_2 \).

Conclusion

- gHSTCP+gARED can provide better performance in terms of fairness and throughput.
  - gHSTCP: two modes in congestion avoidance phase based on the changing trend of RTT
  - gARED: adapt \( \text{max}_{\text{p}} \) according to average queue length and its trend in variation
- Future works: further investigation, e.g.,
  - Recover effectively from simultaneous packet losses
  - The impact on short-lived traffic

Thanks