Dynamic threshold Control of RED for establishing fairness among thousands of TCP connections

Go Hasegawa
Osaka University, Japan
hasegawa@ics.es.osaka-u.ac.jp
Backgrounds

- **Internet Router Buffer**
  - Tail-Drop Router
    - Simple, easy implementation
    - Discard arriving packets when buffer is full
      - Bursty packet loss
    - Poor performance
      - Throughput
      - fairness among connections
  - **RED (Random Early Detection)**
    - Discard incoming packets with a certain probability
**RED (Random Early Detection)**

- **Probability is changed according to average queue length**
- **Avoid buffer overflow, keep queue length low**

![Graph of Packet Discarding Probability vs. Average Queue Length](image)

- Packet Discarding Probability
- Average Queue Length [packets]
- Queue Length [packets]
- Time [sec]

- $max_{th}$
- $min_{th}$
- $max_p$
Researches on RED

- Achieve better performance than Tail-Drop
  - Throughput, Fairness among connections
- Difficult to set control parameters
  - Depends on network condition, # of active connections, ...
Researches on RED (2)

- Many enhanced algorithm have been proposed
  - Introducing new control parameters
  - Parameter setting problem remains
- Focus only on throughput
- Small number of connections
Objectives

- Evaluate fairness of RED with many connections
  - Compare with Tail-Drop
- Propose new algorithm of RED
  - Set threshold values dynamically
  - Easy parameter setting
  - Provide good fairness
Network Model

N sender hosts transmit infinite size of data to receiver host by TCP Reno

Evaluate fairness among mean throughputs of N TCP connections

Measure: Fairness Index

\[ f = \frac{\left( \sum_{i=1}^{N} x_i \right)^2}{N \sum_{i=1}^{N} x_i^2} \]

- 0 (unfair) < f < 1 (completely fair)

Sender Host 1

Sender Host 2

Sender Host N

Router Buffer Size: \( B \) packets

100Mbps, 2msec

1.5Mbps, 4msec

Receiver Host

100Mbps, 2msec

1.5Mbps, 4msec

8th, Nov 2001

APSITT2001
Evaluation Result (1): $B=1000$ [packets]

- Tail-Drop shows degrades fairness when $N$ is small
- Recommended parameter set of RED shows bad fairness especially when $N$ is large
- RED with appropriate parameters shows good fairness regardless of $N$
Evaluation Result (2): $B=10000$ [packets]

- Tail-Drop and RED with recommended parameter set cannot provide fairness
- Best setting of RED changes according to network condition
Fairness of RED

- **Recommended parameter set cannot provide fairness when N is large**
  - When average queue length becomes $max th$, RED discards incoming packets in burst as Tail-Drop

- **Threshold value ($max th$) should be set carefully to keep fairness**

- **Appropriate value will be changed by various factors**
  - Bandwidth, Buffer size, # of connections, delay, ...

- **Static parameter setting is impossible**
Proposed Algorithm: dt-RED

- Observe average queue length at regular intervals
- Change threshold values ($\text{max}_{\text{th}}$, $\text{min}_{\text{th}}$) according to the average queue length
- Keep average queue length between $\text{max}_{\text{th}}$ and $\text{min}_{\text{th}}$
- Avoid bursty packet losses at the RED router buffer
Evaluation Result (3): \( B = 1000 \) [packets]

- \textit{dt-RED} can provide the best fairness among all algorithms
**Evaluation Result (4): B=1000 [packets]**

- **dt-RED shows good fairness regardless of the number of connections and buffer size**
Characteristics of dt-RED

- **Always provide the same fairness as original RED with best parameter set**
  - Automatically set threshold values appropriately

- **Show good performance with one parameter set, regardless of network condition**
  - Remove the difficulty of parameter setting in original RED
Conclusion

- Fairness of RED algorithm is largely dependent on control parameters
- Static parameter setting is difficult
- dt-RED can set parameters appropriately according to dynamic change of network condition