Random Early Detection Gateways for Congestion Avoidance

Sally Floyd and Van Jacobson,

IEEE Transactions on Networking,
Vol.1, No. 4, (Aug 1993), pp.397-413.

Outline

- Introduction
- Background: Definitions and Previous Work
- The RED Algorithm
- RED parameters
- Evaluation of RED
- Conclusions and Future Work

Introduction

- **Main idea**: to provide congestion control at the router for TCP flows.
- **Goals of RED**
  - **[primary goal]** is to provide congestion avoidance by controlling the average queue size such that the router stays in a region of low delay and high throughput.
  - To avoid global synchronization (e.g., in Tahoe TCP).
  - To control misbehaving users (this is from a fairness context).
  - To seek a mechanism that is not biased against bursty traffic.

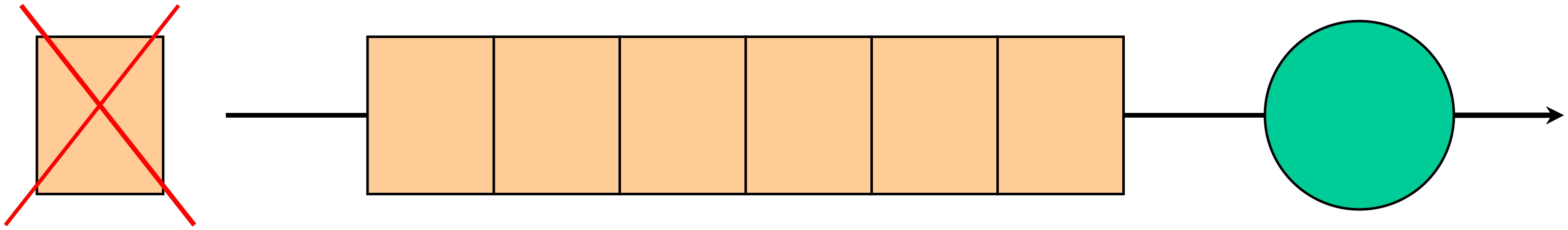
Definitions

- *congestion avoidance* – when impending congestion is indicated take action to avoid congestion
- *incipient congestion* – congestion that is beginning to be apparent.
- need to notify connections of congestion at the router by either *marking* the packet [ECN] or *dropping* the packet {This assumes a drop is an implied signal to the source host.}

Previous Work

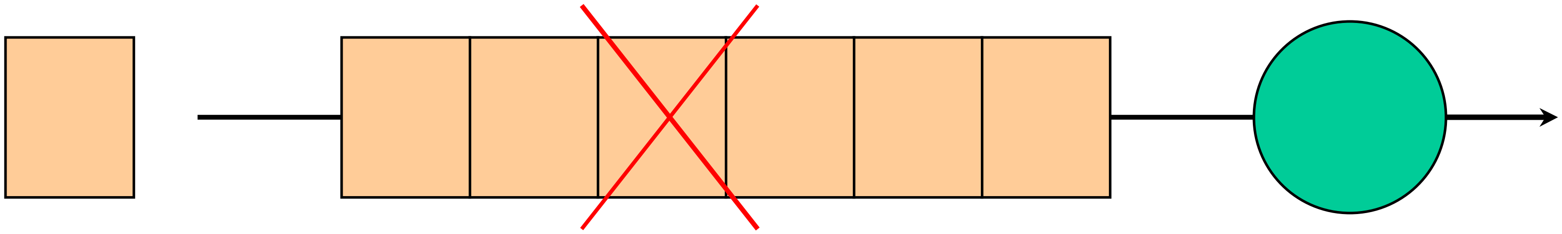
- Drop Tail
- Random Drop
- Early Random Drop
- Source Quench messages
- DECbit scheme

Drop Tail Router



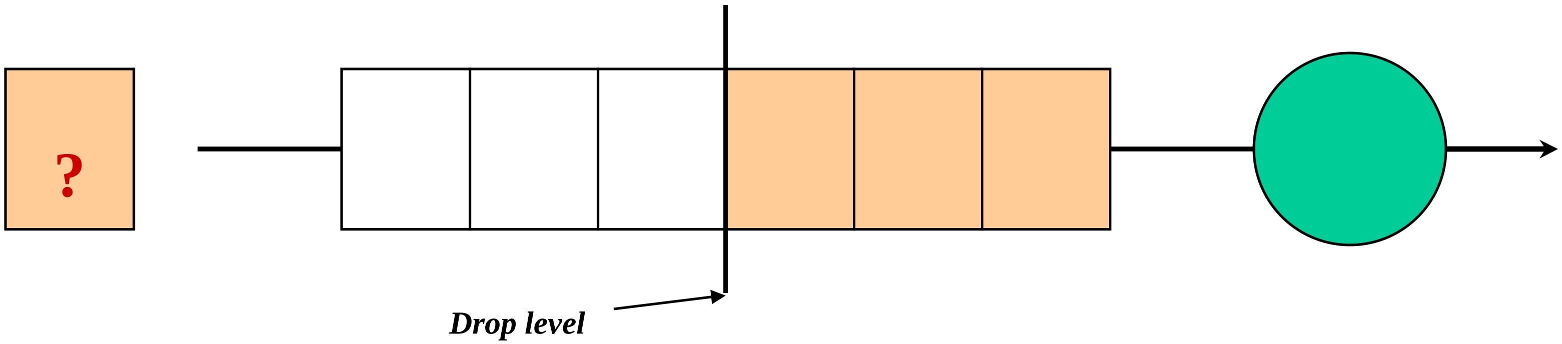
- FIFO queueing mechanism that drops packets when the queue overflows.
- Introduces *global synchronization* when packets are dropped from several connections.

Random Drop Router



- When a packet arrives and the queue is full, randomly choose a packet from the queue to drop.

Early Random Drop Router



- If the queue length exceeds a drop level, then the router drops each arriving packet with a fixed *drop probability*.
- Reduces global synchronization
- Did **not** control misbehaving users (UDP)

Source Quench message

- Router sends *source quench* messages back to source before queue reaches capacity.
- Complex solution that gets router involved in end-to-end protocol.

DECbit scheme

- Uses a *congestion-indication bit* in packet header to provide feedback about congestion.
- Average queue length is calculated for last (busy + idle) period plus current busy period.
- When average queue length exceeds one, set congestion-indicator bit in arriving packet's header.
- If at least half of packets in source's last window have the bit set, then decrease the window exponentially.

RED Algorithm

for each packet arrival

calculate the average queue size *avg*

if $min_{th} <= avg < max_{th}$

calculate the probability p_a

with probability p_a:

mark the arriving packet

else if $max_{th} <= avg$

mark the arriving packet

RED drop probability (p_a)

$$p_b = max_p \text{ x } (avg - min_{th})/(max_{th} - min_{th}) \quad [1]$$

where

$$p_a = p_b / (1 - count \text{ x } p_b) \quad [2]$$

Note: this calculation assumes queue size is measured in packets. If queue is in bytes, we need to add [1.a] between [1] and [2]

$$p_b = p_b \text{ x PacketSize/MaxPacketSize} \quad [1.a]$$

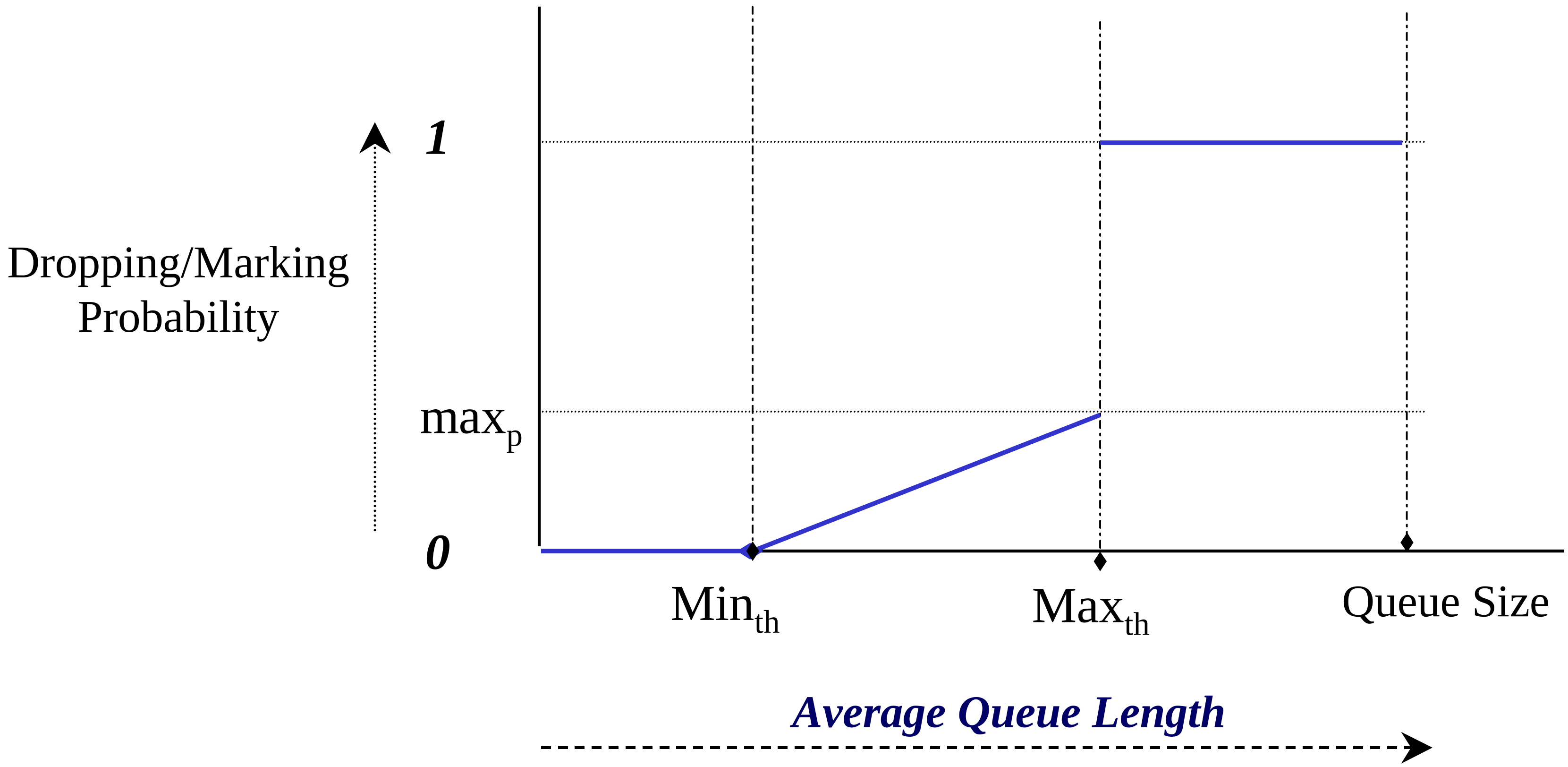
average queue length (*avg*)

$$avg = (1 - w_q) \times avg + w_q \times q$$

where q is the newly measured queue length

This exponential weighted moving average is designed such that short-term increases in queue size from bursty traffic or transient congestion do not significantly increase average queue size.

RED/ECN Router Mechanism



RED parameter settings

- w_q suggest 0.001 <= w_q <= 0.0042

  authors use w_q = 0.002 for simulations
- min_{th}, max_{th} depend on desired average queue size
  - bursty traffic ➔ increase min_{th} to maintain link utilization.
  - max_{th} depends on maximum average delay allowed
  - RED most effective when average queue size is larger than typical increase in calculated queue size in one round-trip time
  - *"rule of thumb"*: max_{th} at least twice min_{th} . However, max_{th} = 3 times min_{th} some experiments shown.

packet-marking probability

- goal: want to uniformly spread out *marked* packets - this reduces global synchronization.
- **Method 1: geometric random variable**
  - each packet marked with probability p_b
- **Method 2: uniform random variable**
  - marking probability is $p_b / (1 - count \times p_b)$ where *count* is the number of unmarked packets arrived since last marked packet.

Figure 8 Here

max_p

- RED performs best when packet-marking probability changes fairly slowly as the average queue size changes
- Recommend that max_p never greater than 0.1

Figure 4 and 5 Here

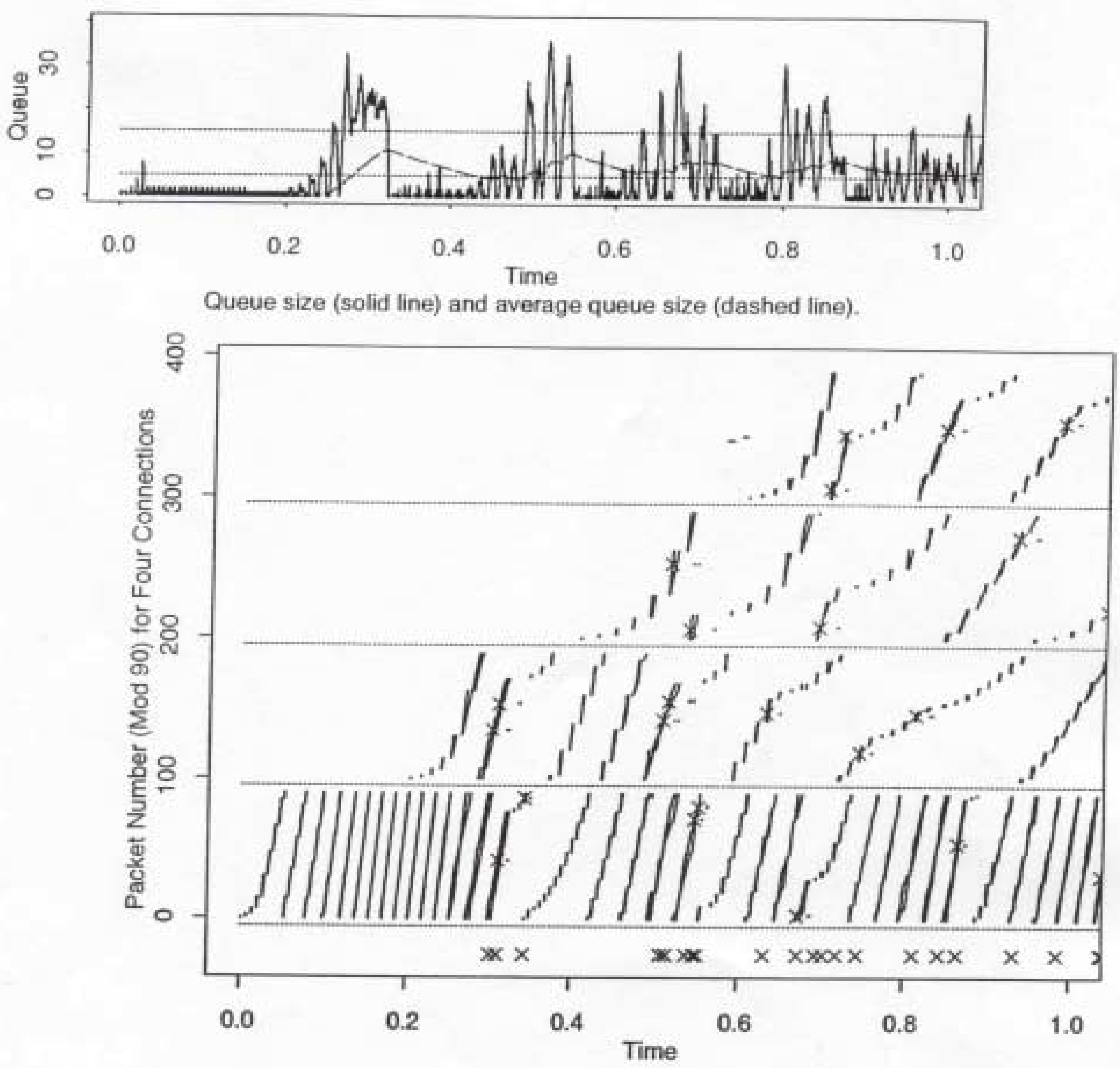


Figure 3: A simulation with four FTP connections with staggered start times.

Figure 6-13 Here

ACN: RED paper

Evaluation of RED meeting design goals

- congestion avoidance
  - If RED *drops* packets, this guarantees the calculated average queue size does not exceed the max threshold. If w_q set properly RED controls *actual* average queue size.
  - If RED *marks* packets, router relies on source cooperation to control average queue size.

Evaluation of RED meeting design goals

- appropriate time scales
  - detection time scale *roughly matches* time scale of response to congestion
  - RED does not notify connections during transient congestion at the router

Evaluation of RED meeting design goals

- no global synchronization
  - avoid global synchronization by marking at as low a rate as possible with distribution spread out
- simplicity
  - detailed argument about how to cheaply implement in terms of adds and shifts

Evaluation of RED meeting design goals

- maximizing global power
  - *power defined as ratio of throughput to delay*
  - see Figure 5 for comparision against drop tail
- fairness
  - authors claim not well-defined
  - {obvious side-step of this issue}
  - [becomes **big deal -**see FRED paper]

Conclusions

- RED is effective mechanism for congestion avoidance at the router in cooperation with TCP.
- *claim:* probability that RED chooses a particular connection to notify during congestion is roughly proportional to that connection's share of the bandwidth.

Future Work (circa 1993)

- Is RED really fair?
- How do we tune RED?
- Is there a way to optimize power?
- What happens with other versions of TCP?
- How does RED work when mixed with drop tail routers?
- How robust is RED?
- What happens when there are many flows?