

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.

Presented by Bob Kinicki



Outline

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



Introduction

Main idea :: *to provide congestion control at the router for TCP flows.*

- **RED Algorithm Goals**

- The **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

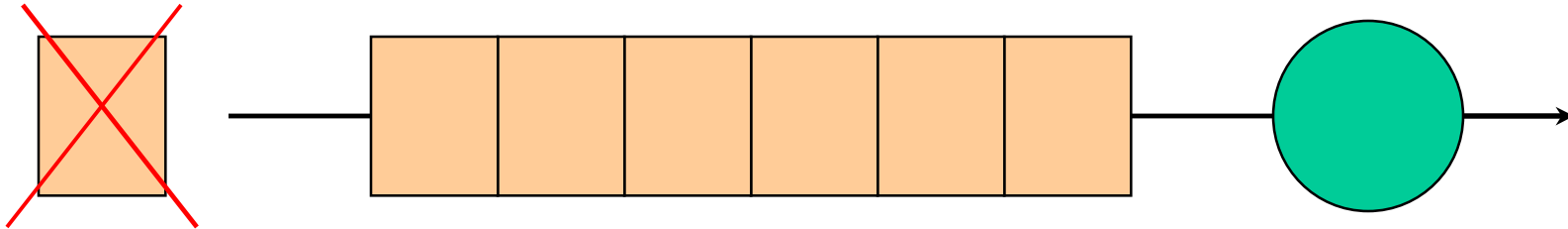
- *congestion avoidance* – when impending congestion is indicated, take action to avoid congestion.
- *incipient congestion* – congestion that is beginning to be apparent.
- need to notify connections of congestion at the router by either *marking* the packet [ECN] or *dropping* the packet { This assumes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

Previous Work

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

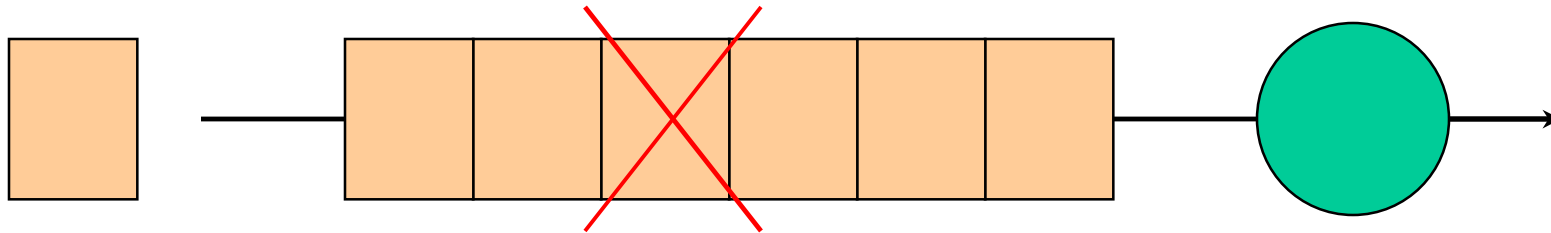


Drop Tail Router



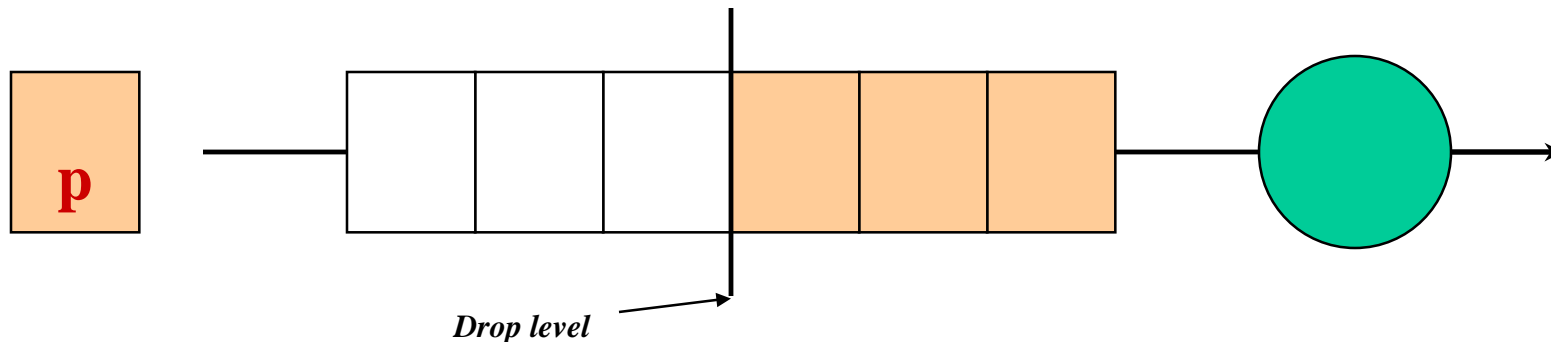
- FIFO queuing mechanism that drops packets from the tail when the queue overflows.
- Introduces *global synchronization* when packets are dropped from several connections.

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
- Does **not** control misbehaving users (UDP)

Source Quench messages

- Router sends *source quench messages* back to source before the 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.
- Upon packet arrival, the average queue length is calculated for last (busy + idle) period plus current busy period.
- When the average queue length exceeds one, the router sets the congestion-indicator bit in arriving packet's header.
- If at least half of packets in source's last window have the bit set, decrease the congestion window exponentially.

RED Algorithm

for each packet arrival

calculate the average queue size avg

if $min_{th} \leq avg < max_{th}$

calculate the probability p_a

with probability p_a :

mark the arriving packet

else if $max_{th} \leq avg$

mark the arriving packet.



RED drop probability (p_a)

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

where

$$p_a = p_b / (1 - count \times p_b) \quad [2]$$

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]

$$p_b = p_b \times PacketSize / MaxPacketSize \quad [1.a]$$

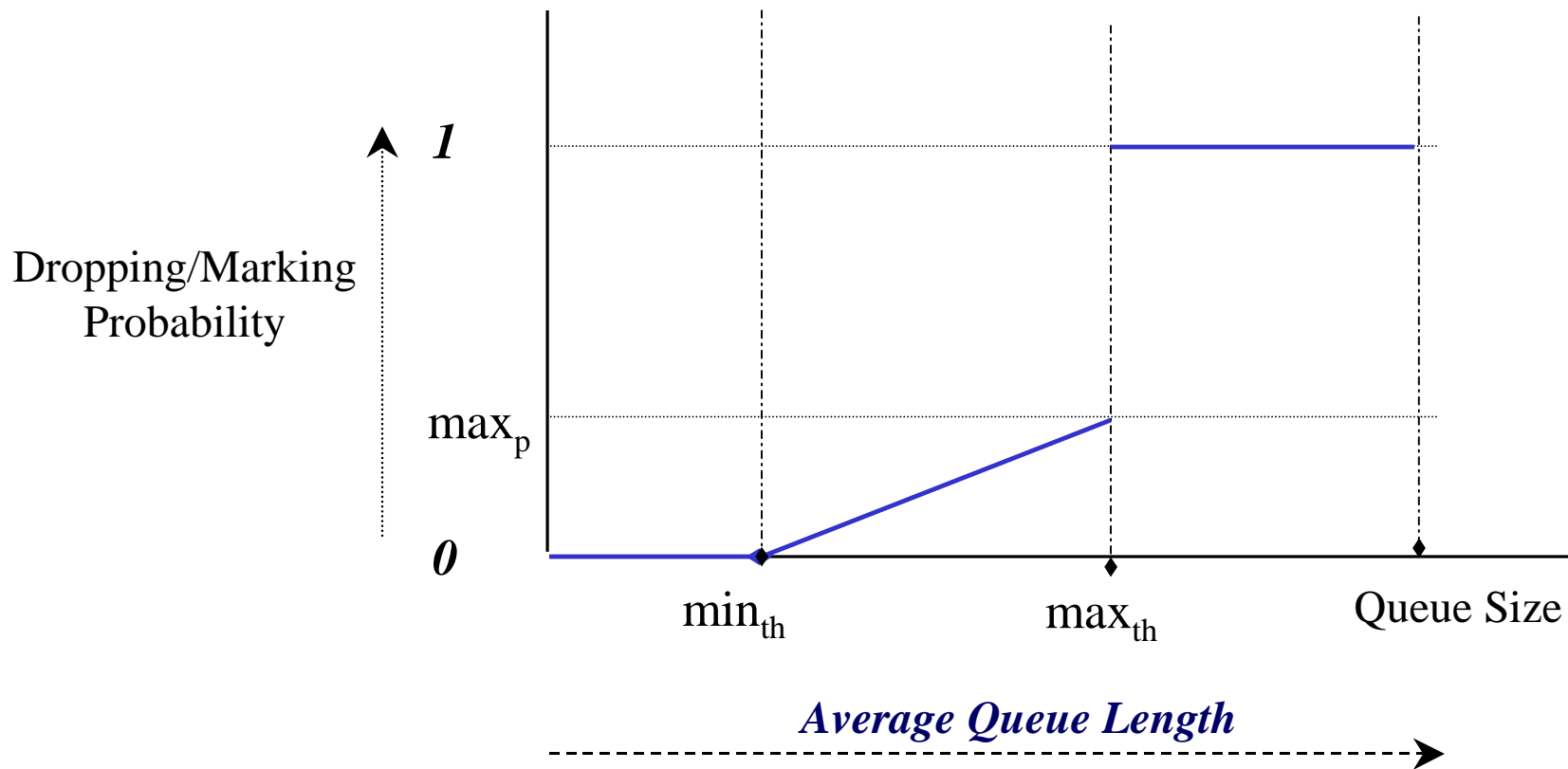
avg - average queue length

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

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 $0.001 \leq w_q \leq 0.0042$
authors use $w_q = 0.002$ for simulations
- min_{th} , max_{th} depend on desired average queue size
 - bursty traffic \rightarrow increase min_{th} to maintain link utilization.
 - max_{th} depends on the maximum average delay allowed.
 - **RED** is most effective when $max_{th} - min_{th}$ is larger than typical increase in calculated average queue size in one round-trip time.
 - **“parameter setting rule of thumb”**: max_{th} at least twice min_{th} . However, $max_{th} = 3$ times min_{th} is used in some of the experiments shown.

packet-marking probability

- The goal is to uniformly spread out the *marked* packets. This reduces global synchronization.

Method 1: geometric random variable

When each packet is marked with probability p_b , the packet intermarking time, X , is a geometric random variable with $E[X] = 1/p_b$.

- This distribution will both cluster packet drops and have some long intervals between drops!!

packet-marking probability

Method 2: uniform random variable

Mark packet with probability $p_b / (1 - count \times p_b)$ where *count* is the number of unmarked packets that have arrived since last marked packet.

$$E[X] = 1/(2 p_b) + 1/2$$



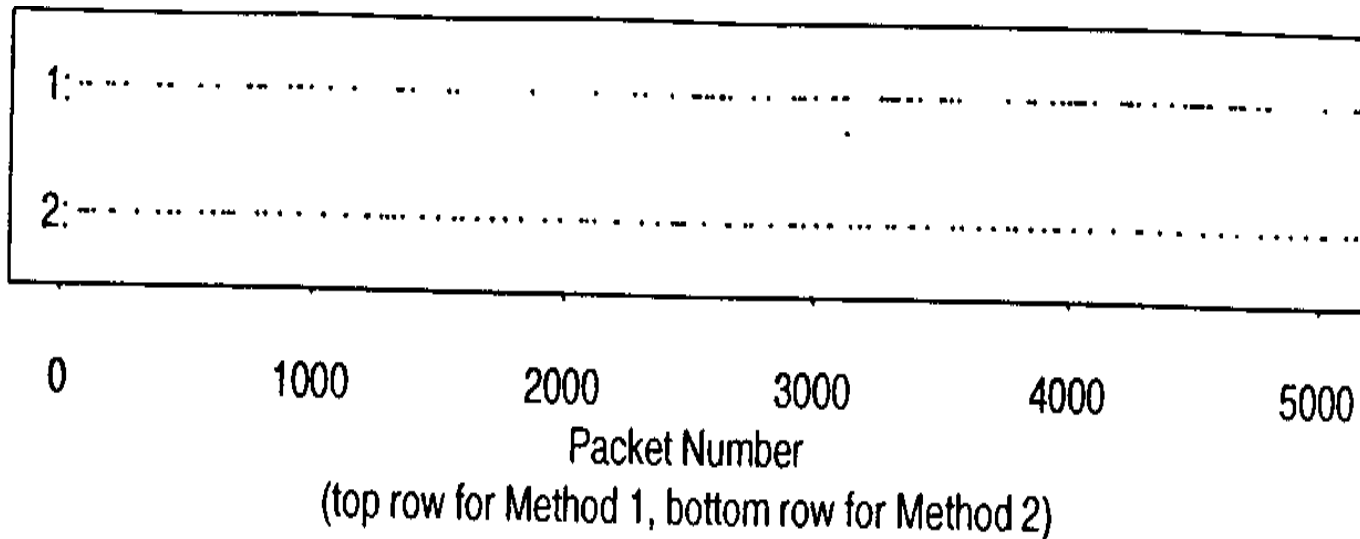


Figure 8: Randomly-marked packets, comparing two packet-marking methods.

Method 1: geometric $p = 0.02$

Method 2: uniform $p = 0.01$

Result :: *marked packets more clustered for method 1*
 → *uniform is better at eliminating “bursty drops”*

Setting max_p

- “**RED** performs best when packet-marking probability changes fairly slowly as the average queue size changes.”
 - This is a **stability argument** in that the claim is that RED with small max_p will reduce oscillations in *avg* and actual marking probability.
- They recommend that max_p never be greater than 0.1

{ This is not a robust recommendation. }



RED Simulations

- Figure 4: Four heterogeneous FTP sources
- Figure 6: Two homogeneous FTP sources
- Figure 10: 41 Two-way, short FTP and TELNET flows
- Figure 11: Four FTP non-bursty flows and one bursty FTP flow

Simple Simulation Four Heterogeneous FTP Sources

TCP Tahoe
1KB packet size
 $w_q = 0.002$
 $max_p = 1/50$
 $min_{th} = 5$
 $max_{th} = 15$
 $max_{cwnd} = bdp$
Large Buffer

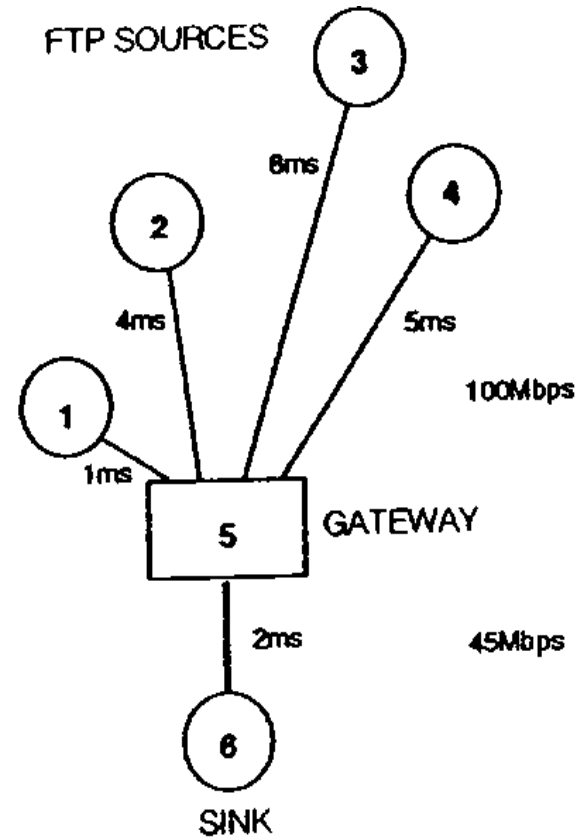
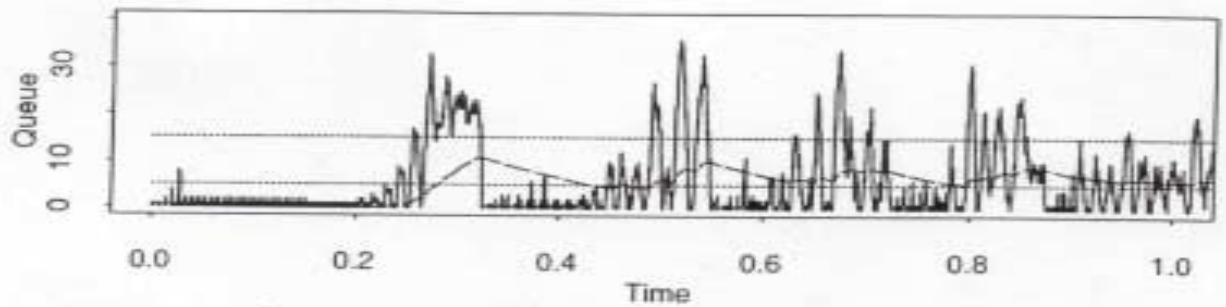


Figure 4: Simulation network.



Queue size (solid line) and average queue size (dashed line).

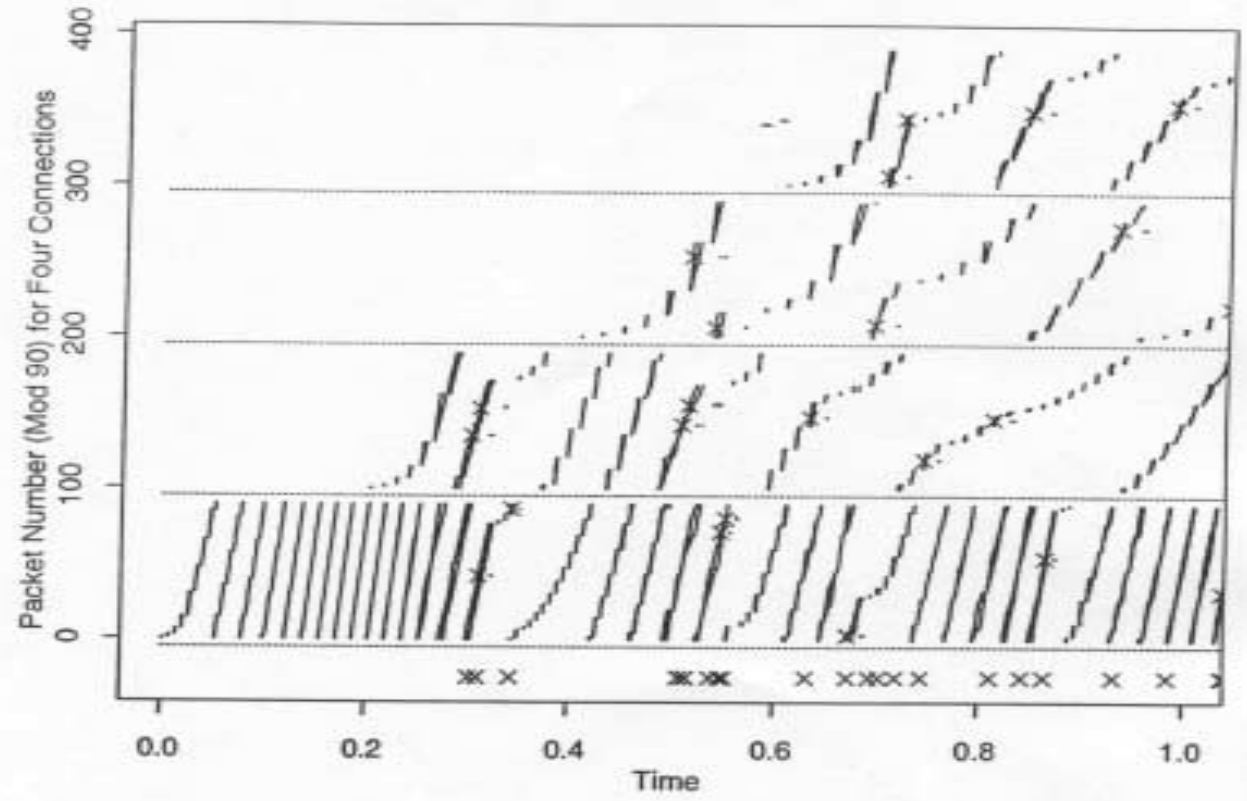


Figure 3: A simulation with four FTP connections with staggered start times.



Two Homogeneous FTP Sources

- RED varies min_{th} from 3 to 50 packets
- Drop Tail varies buffer from 15 to 140 packets
- max cwnd = 240 packets

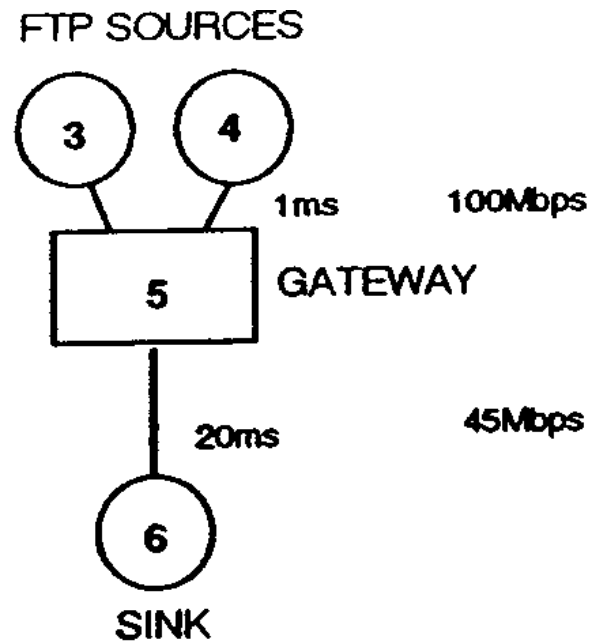


Figure 6: Simulation network.

Two Homogeneous FTP Sources

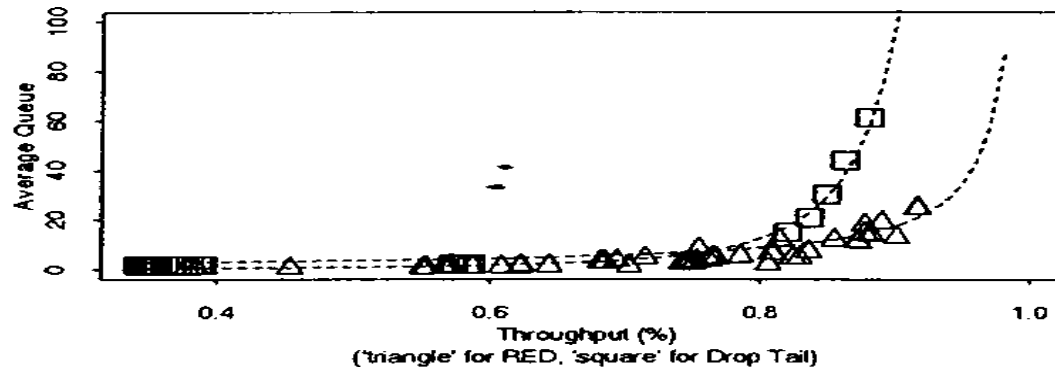


Figure 5: Comparing Drop Tail and RED gateways.

Figure 5 represents many simulation experiments.

RED yields lower queuing delay as utilization improves by increasing min_{th} from 3 to 50 packets.

Drop-tail yields *unacceptable delay* at high utilization.

The power measure is better for **RED** !

Network with 41 Short Duration Connections

Two-way traffic of FTP and TELNET traffic .

Total number of packets per Connection varies from 20 To 400 packets.

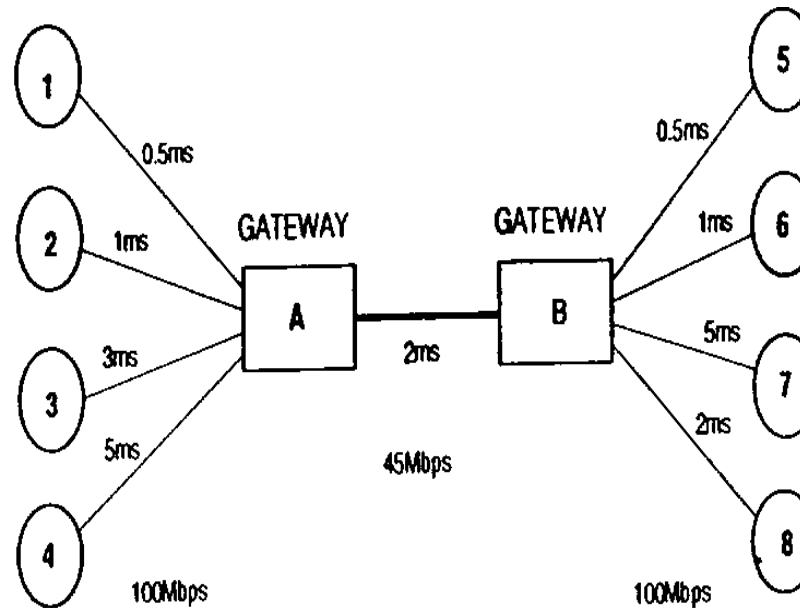


Figure 10: A network with many short connections.

Short, two-way FTP and TELNET flows

- **RED** controls the average queue size.
- Flows have small **cwnd** **maximums (8 or 16)**.
- Packet dropping is higher and bursty.
- Utilization is low (**61%**).
- Mentions **ACK-compression** as one cause of bursty packet arrivals.

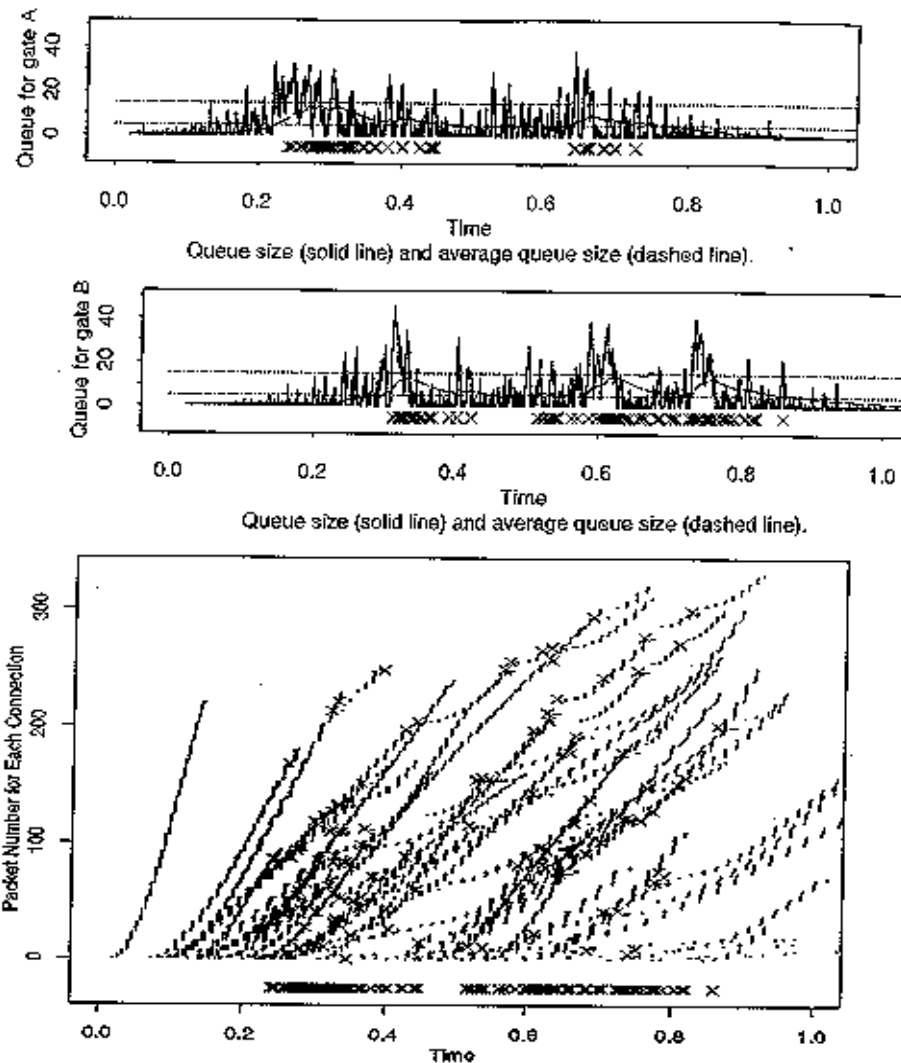


Figure 9: A RED gateway simulation, with heavy congestion, two-way traffic, and many short FTP and TELNET connections.

Five FTP Flows Including One Bursty Flow

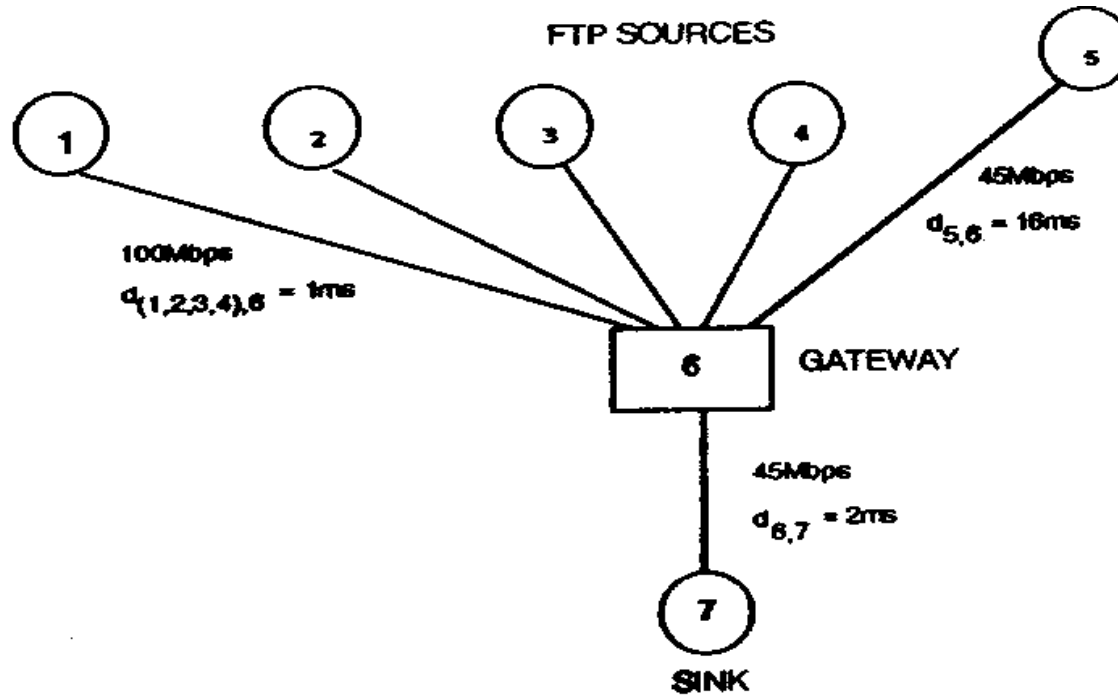


Figure 11: A simulation network with five FTP connections.

Simulation Details

- Bursty traffic == large RTT, small cwnd
- Other traffic = small RTT, small cwnd
{robust flows}
- Node 5 :: the bursty flow cwnd = 8 packets
- Each simulation run for 10 seconds and each mark in figures represents one second.

Drop Tail Gateways

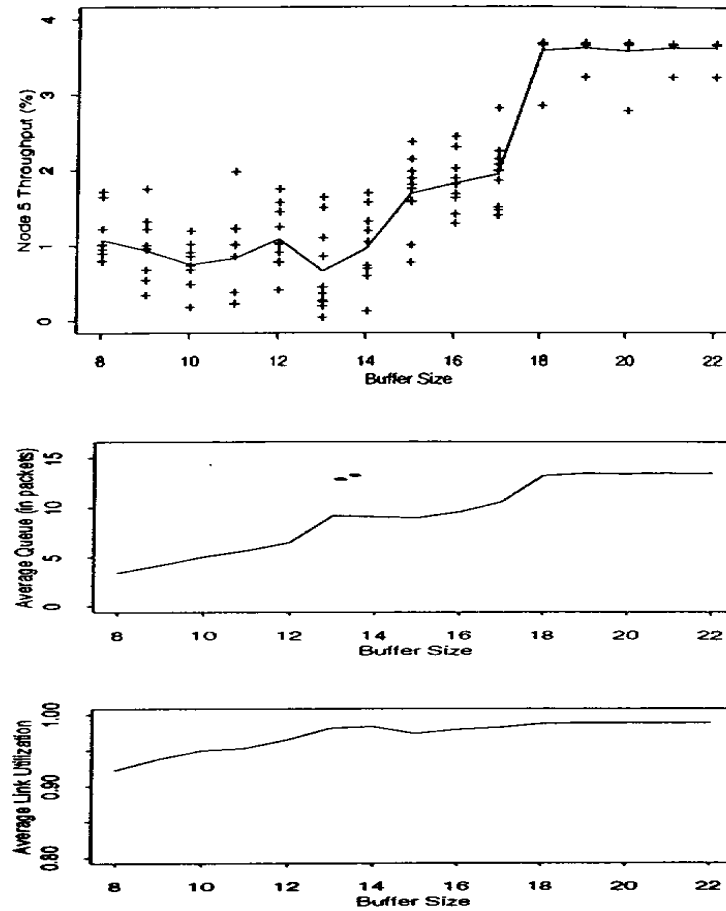


Figure 12: Simulations with Drop Tail gateways.

Random Drop Gateways

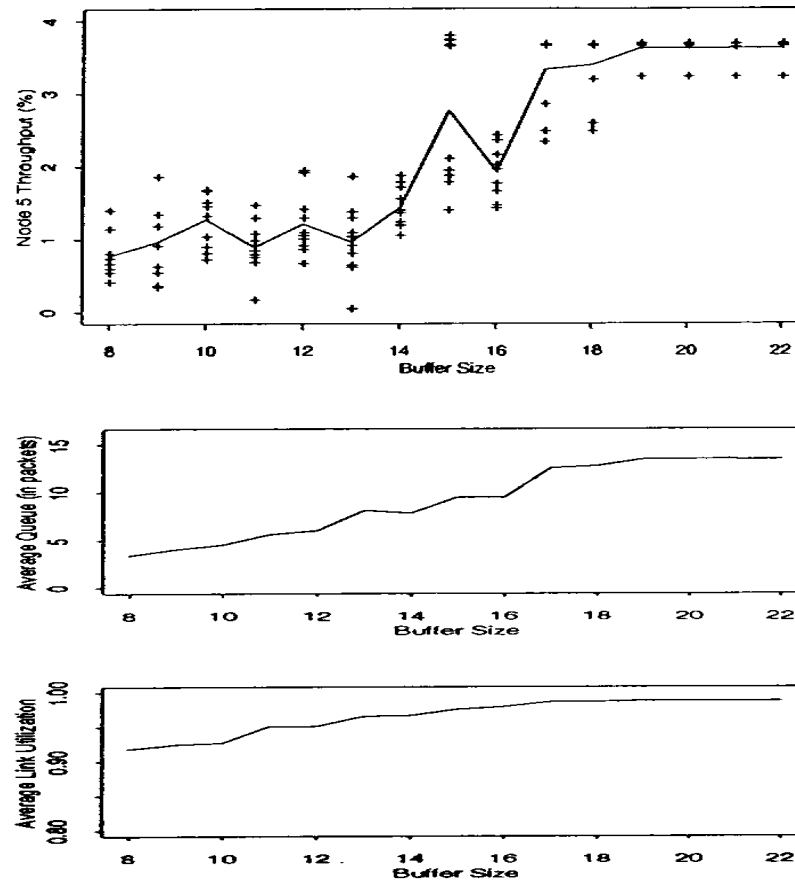


Figure 13: Simulations with Random Drop gateways.



RED Gateways

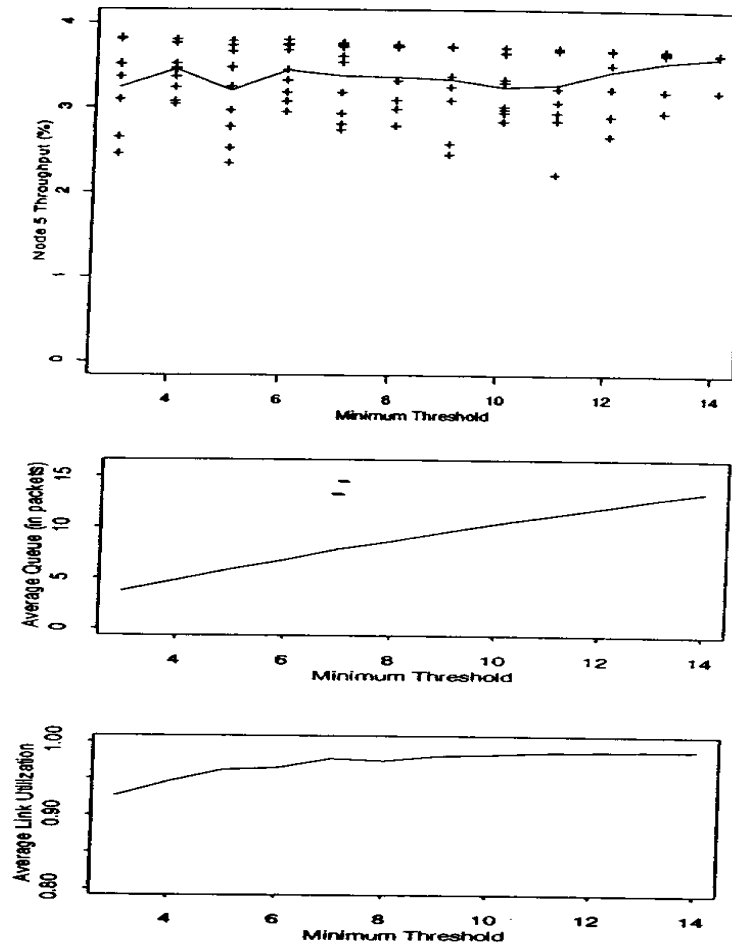


Figure 14: Simulations with RED gateways

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 is set properly, RED controls the *actual* average queue size.
 - If RED *marks* packets when *avg* exceeds max_{th} , the router relies on source cooperation to control the average queue size. {not part of RED}

Evaluation of **RED** meeting design goals

- appropriate time scales
 - **claim**:: The 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
 - RED avoids global synchronization by marking at as low a rate as possible with marking distribution spread out.
- simplicity
 - detailed argument about how to cheaply implement in terms of adds and shifts.
- {Historically, the *simplicity of RED* has been strongly refuted because RED has too many parameters to make it robust. }



Evaluation of RED meeting design goals

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



Conclusions

- **RED** is effective mechanism for congestion avoidance at the router in cooperation with TCP.
- **claim::** The 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?