Random Early Detection Gateways for Congestion Avoidance

Sally Floyd and Van Jacobson

IEEE/ACM Trans. Networking, 1993

Content

- Why gateway congestion control ?
- Previous work on congestion avoidance gateway
- Guidelines and Principles of the RED algorithm
- Implementation issues
- Simulation results

Why gateway congestion control?

- More effective congestion detection
- Higher throughput and low queue size
- Not all flows are cooperative

Previous work on congestion avoidance gateways

- Drop Tail
- Random Drop or Early Random Drop
- IP Source Quench
- DECbit (Congestion Indication)

Drop Tail

• packets are dropped when the queue overflows

• suffer from global synchronization (loss in throughput)

Random Drop

• randomly choose a packet from the queue to drop when the queue is full

• drop probability is proportional to transmission rate

• unfair for different round-trip time paths

Early Random Drop

- drop each packet arriving at the gateway with a fixed drop probability when the queue length exceeds a certain drop level
- unsuccessful in controlling misbehaving users
- scaling problem with multiples congested gateways

IP Source Quench

- send an ICMP source quench to the source to indicate congestion
- discrepancy in implementations

DECbit (Congestion Indication)

- calculate the average queue length during last busy+idle period plus the current busy period
- set the congestion indication bit in the header when the average queue length exceeds 1
- if at least half of packets in the last windows has the congestion bit set, the windows is decreased exponentially; otherwise, the windows is increased linearly.
- variations in period of queue-size computation
- not TCP compatible

Random Early Detection (1)

benefits:

- provide both congestion recovery and congestion avoidance
- avoid global synchronization and biases against bursty traffic
- maintain an upper bound of average queue size
- work with TCP and non-TCP transport-layer protocol

principles:

- monitor the average queue size for each output queue
- use randomization to choose to notify congestion
- accommodate both transient and longer-lived congestion
- probability to be notified is proportional to share

Random Early Detection (2)

algorithm:

```
For each packet arrival
    calculate the average queue size avg
    if min_th <= avg < max_th
      calculate probability p_a
      with probability p_a:
           mark the arriving packet
    else if max_th <= avg
           mark every arriving packet
```

Random Early Detection (3)

Calculating the average queue length

non-empty queue : avg ← $(1-w_q)avg + w_q q$

empty queue : avg ← $(1-w_q)^m avg$

An upper bound of w_q

rule: accommodate a burst of L packets

avg_L: sum($iw_q(1-w_q)^{L-i}$, i, 1, L) < min_{th}

A lower bound of w_q

rule: respond quickly enough to reflect actual queue size

$-1/\ln(1-w_q)$ packets are required to reflect a steady queue size of 1 packet (reach 1-1/e = 0.63)

Random Early Detection (4)

Setting min_{th} and max_{th}

min_{th} : must be large enough to accommodate busty traffic

max_{th} : must not lead to long average delay

rule of thumb : set max_{th} at least twice min_{th}

Calculating the packet-marking probability

version 1 : $p_b \leftarrow max_p(avg-min_{th})/(max_{th}-min_{th})$

Inter-marking time is geometrically distributed !!

version 2: $p_b \leftarrow p_b/(1-count*p_b)$

where count is # of unmarked packets since the last marked packet

Inter-marking time is uniformly distributed !!

Random Early Detection (5)

The Program

as if m packets had arrived

$C_1 = max_p/(max_{th}-min_{th})$

$C_2 = max_p min_{th}/(max_{th}-min_{th})$

Inter-marking time uniformly distributed between 1 to $1/p_b$

```
Initialization:
   avg ← 0
   count ← −1
for each packet arrival:
 calculate the new average queue size avg:
      if the queue is nonempty
         avg ← avg + w_q (q − avg)
      else using a table lookup:
         avg ← (1 − w_q)^((time−q_time)/s) avg
   if min_th ≤ avg < max_th
      increment count
      p_b ← C_1 · avg − C_2
      if count > 0 and count ≥ Approx[R/p_b]
         mark the arriving packet
         count ← 0
      if count = 0 (choosing random number)
         R ← Random[0, 1]
   else if max_th ≤ avg
      mark the arriving packet
      count ← −1
   else count ← −1
when queue becomes empty
   q_time ← time
New variables:
   R: a random number
New fixed parameters:
   s: typical transmission time
```

Fig. 17. Efficient algorithm for RED gateways.

Simulation (1)

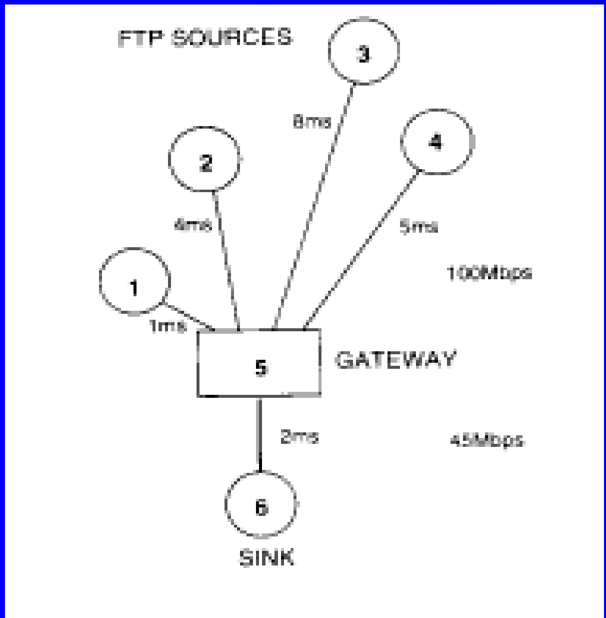


Fig. 4. Simulation network.

Link utilization:

4500*8k / 45M = 80%

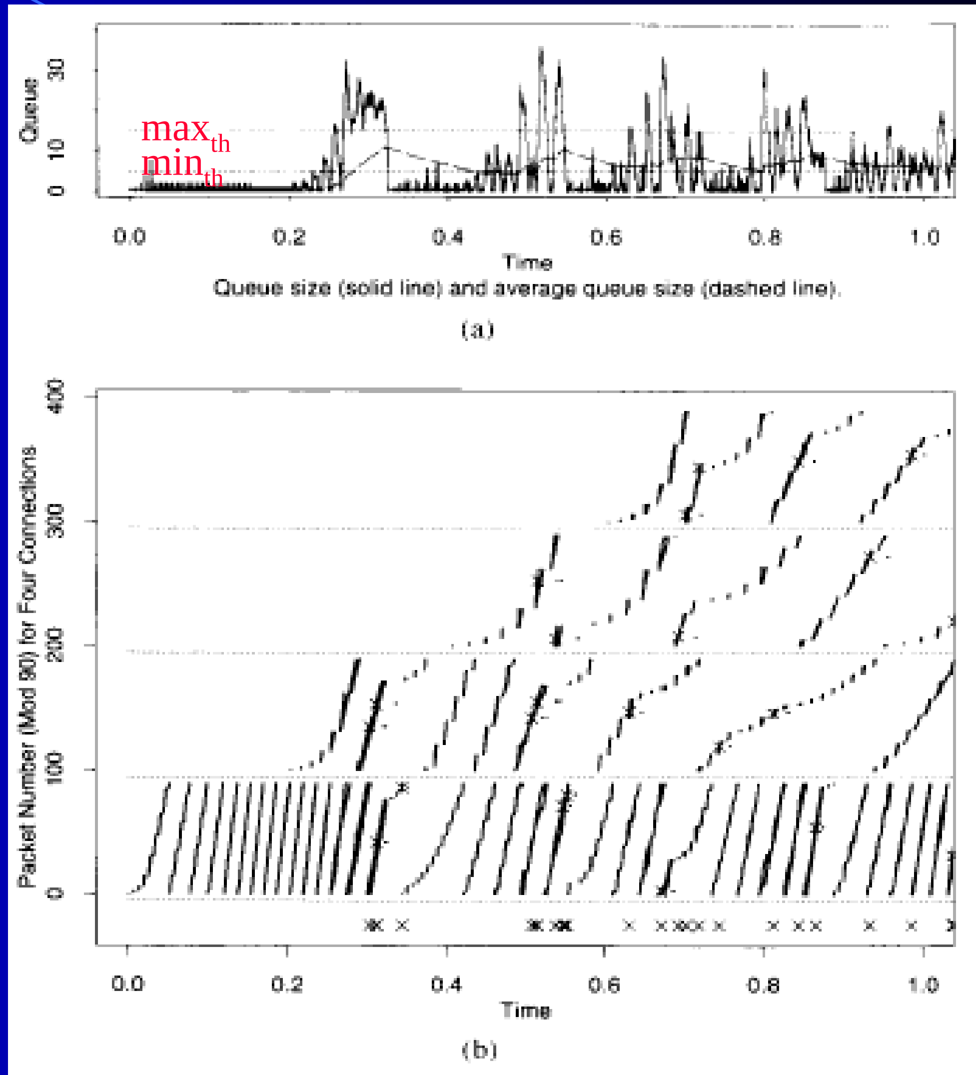
Session 4

Session 3

Session 2

Session 1

X → drop



Simulation (2)

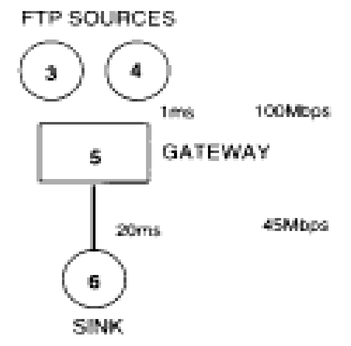


Fig. 6. Simulation network.

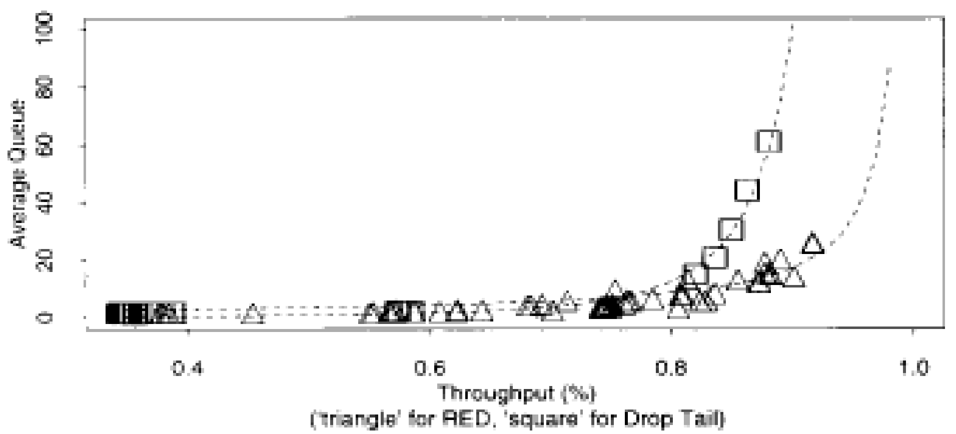


Fig. 5. Comparing Drop Tail and RED gateways.

Comments:

- With the same throughput, RED gateways have shorter average queue size, which means shorter average delay.
- RED gateways have better network power (throughput/delay).
- Drop Tail gateways with small queue drop packets during slow-start phase.
- Drop Tail gateway suffer from global synchronization.

Simulation (3)

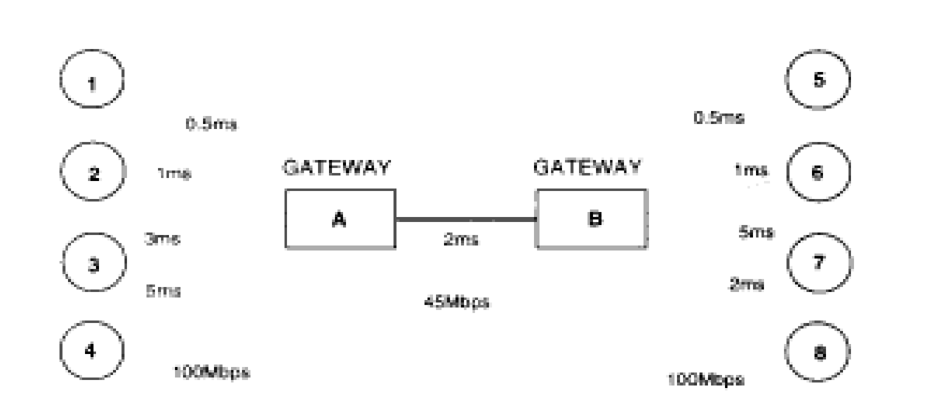
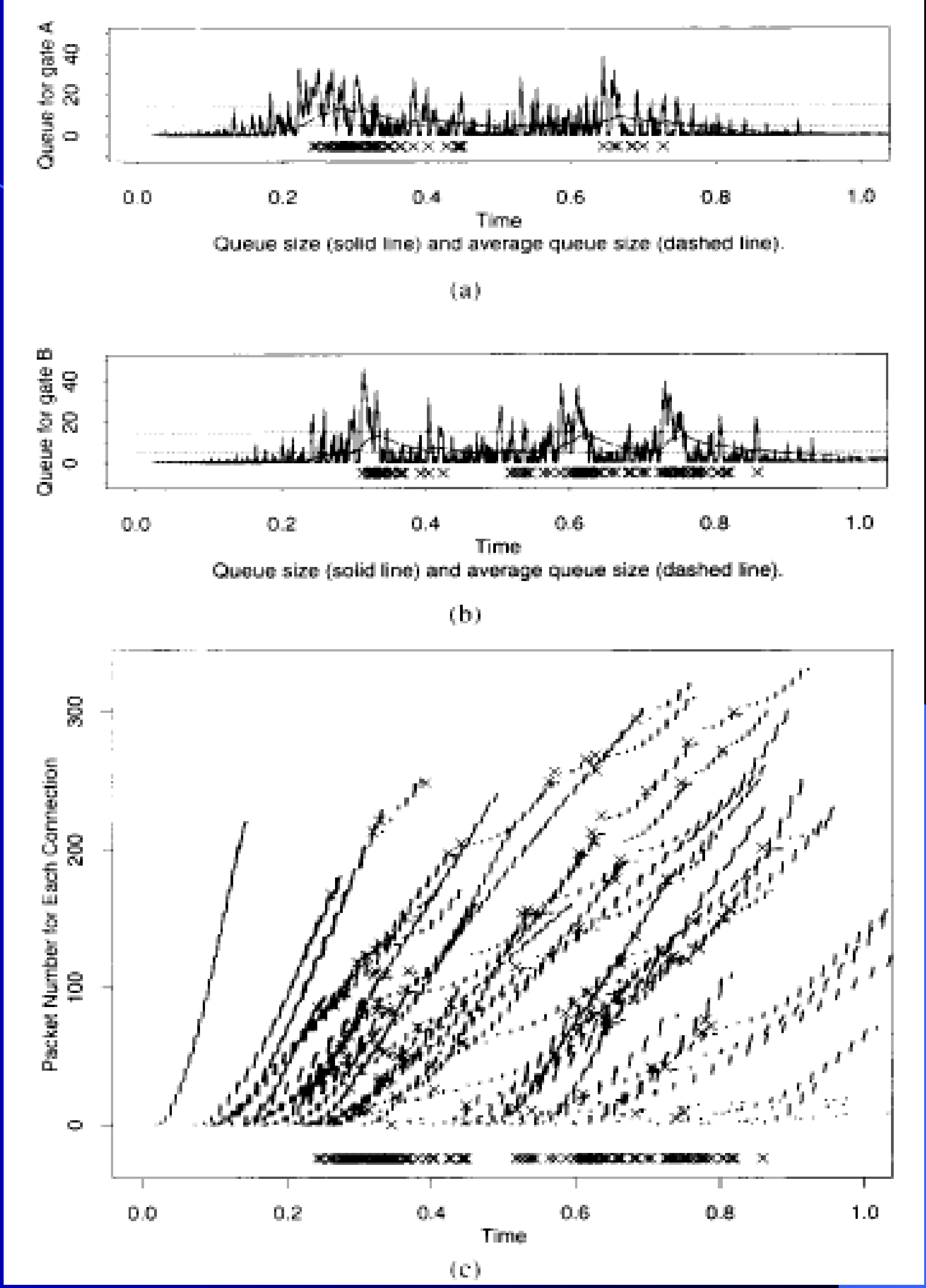
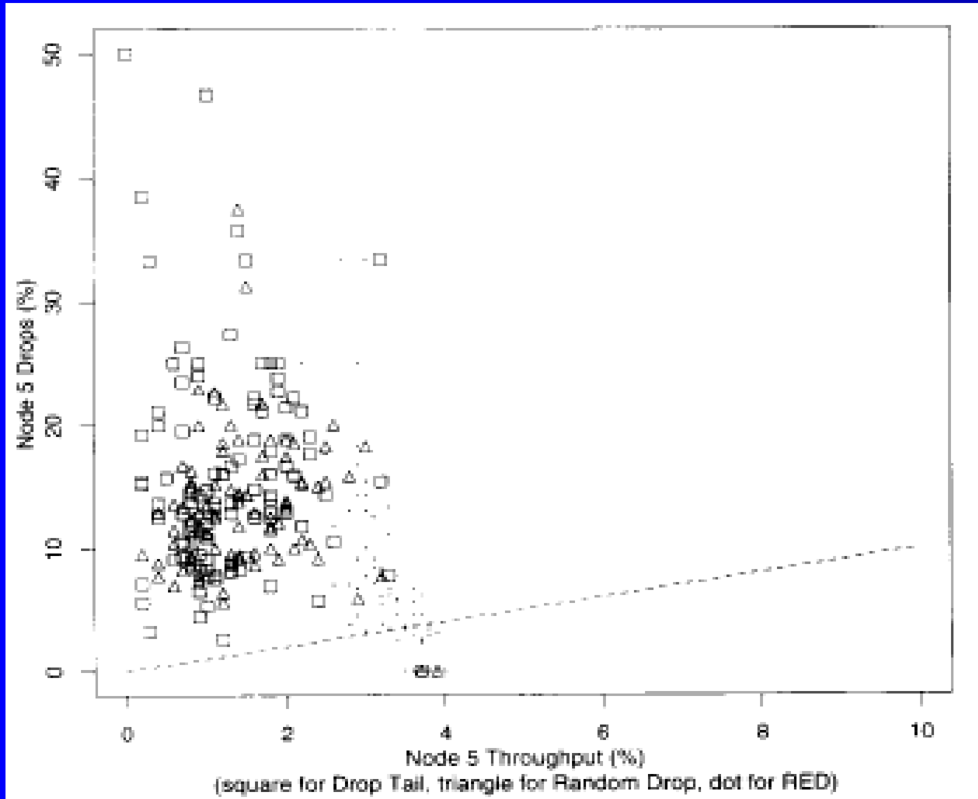


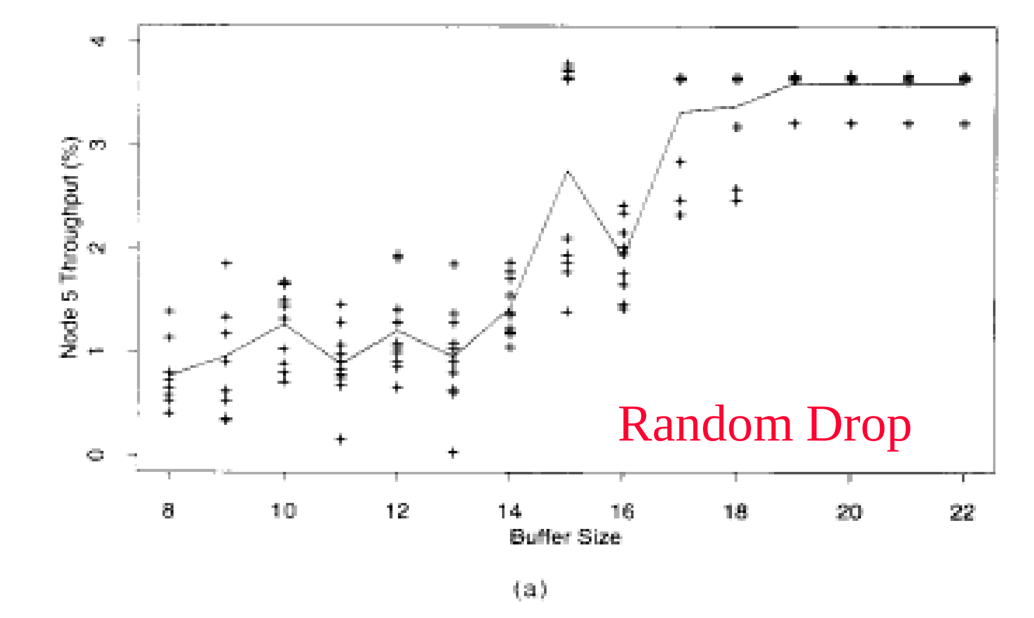
Fig. 10. A network with many short connections.

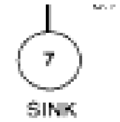


- Control average queue size well !!
- Marking rate reflects the degree of congestion.

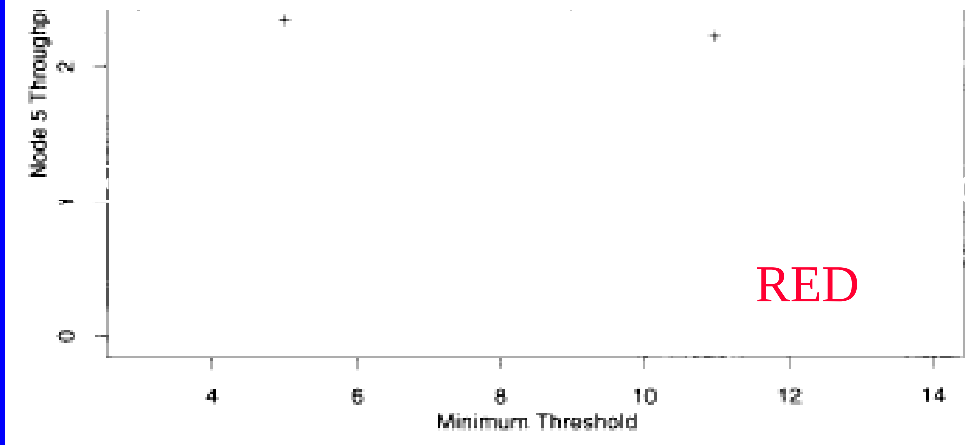
Simulation (4)







with five FTP connections.

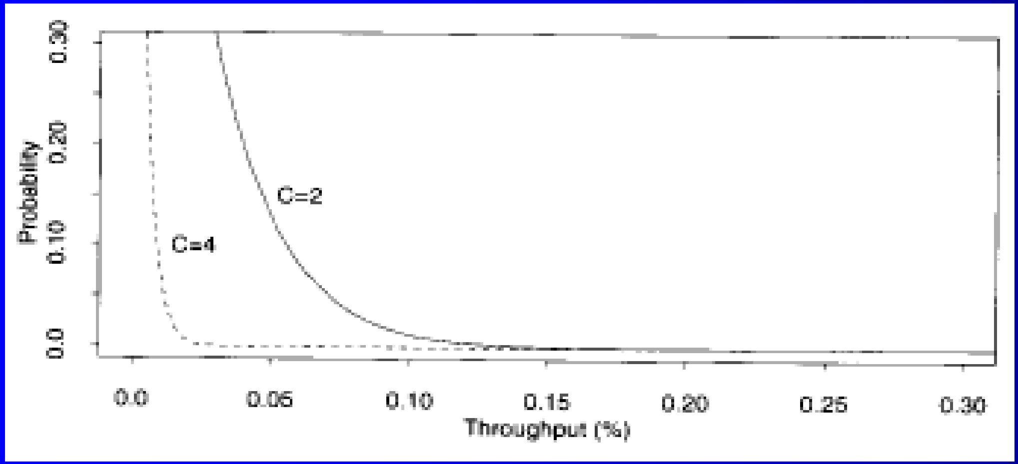


lom Drop has a bias against bursty traffic.

share of the throughput.

Identifying misbehaving users

Let $S_{i,n}$ be the number of the n most-recently-marked packets that come from connection i and the connection i has a fraction pi of the bandwidth. We have $P\{S_{i,n} >= c p_i n\} <= e^{(-2n)(c-1)^2 p_i^2}$



• The connection that has a large fraction of the marked packets is likely to have a large fraction of the average bandwidth.

• So we can identify such a connection with the fraction of marked packets.

Future Work

- optimum average queue size for maximizing throughput and minimizing delay
- traffic dynamics with a mix of Drop Tail and RED gateways
- the RED gateway behavior with transport protocols other than TCP
- priority by marking information in the RED algorithm

More RFCs

1. “Gateway Congestion Control Survey”, RFC 1254
2. “Recommendations on Queue Management and Congestion Avoidance in the Internet”, RFC 2309
3. “IETF Criteria for Evaluated Reliable Multicast Transport and Application Protocols”, RFC 2357
4. “A proposal to add Explicit Congestion Notification to IP”, RFC 2481
5. “TCP Congestion Control”, RFC 2581
6. “Congestion Control Principles”, RFC 2914