The UNIVERSITY *of* NORTH CAROLINA
at CHAPEL HILL

Differential Congestion Notification: Taming the Elephants

Long Le, Jay Aikat, Kevin Jeffay, and Don Smith

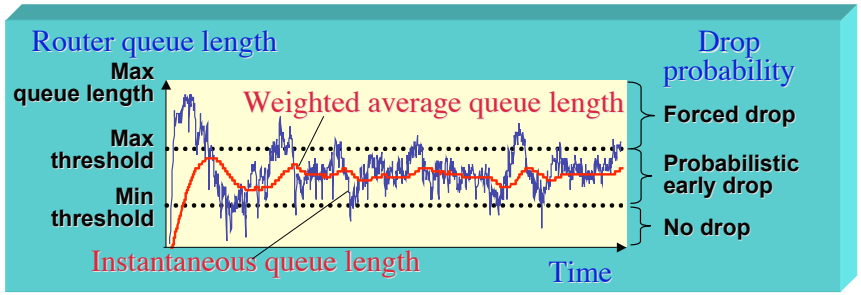
IEEE ICNP 2004

http://www.cs.unc.edu/Research/dirt

Outline

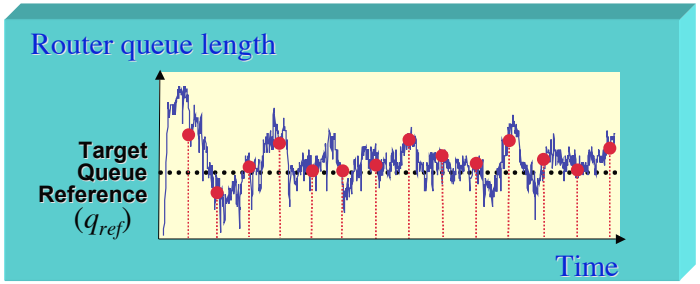
- Background: Router-based congestion control
  - Active Queue Management (AQM)
  - Explicit Congestion Notification (ECN)
- Do AQM schemes work?
- The case for *differential congestion notification* (DCN)
- A DCN prototype and its empirical evaluation

Active Queue Management
The RED Algorithm [Floyd & Jacobson 93]



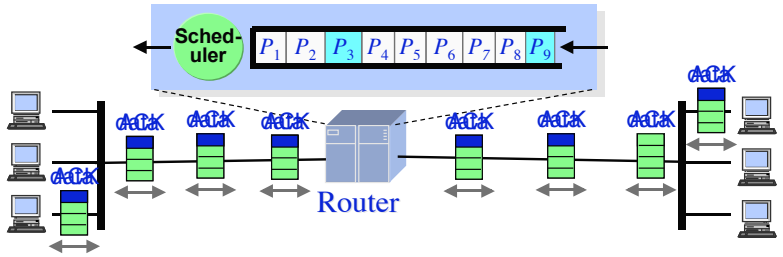
- RED computes a weighted moving average of queue length to accommodate bursty arrivals
- Drop probability is a function of the current average queue length
  - The larger the queue, the higher the drop probability

The Proportional Integral (PI) Controller



- PI attempts to maintain an explicit target queue length
- PI samples instantaneous queue length at fixed intervals and computes a mark/drop probability at k^{th} sample:
  - $p(kT) = a \times (q(kT) - q_{ref}) - b \times (q((k\text{-}1)T) - q_{ref}) + p((k\text{-}1)T)$
  - a, b, and T depend on link capacity, maximum RTT and the number of flows at a router

Explicit Congestion Notification Overview

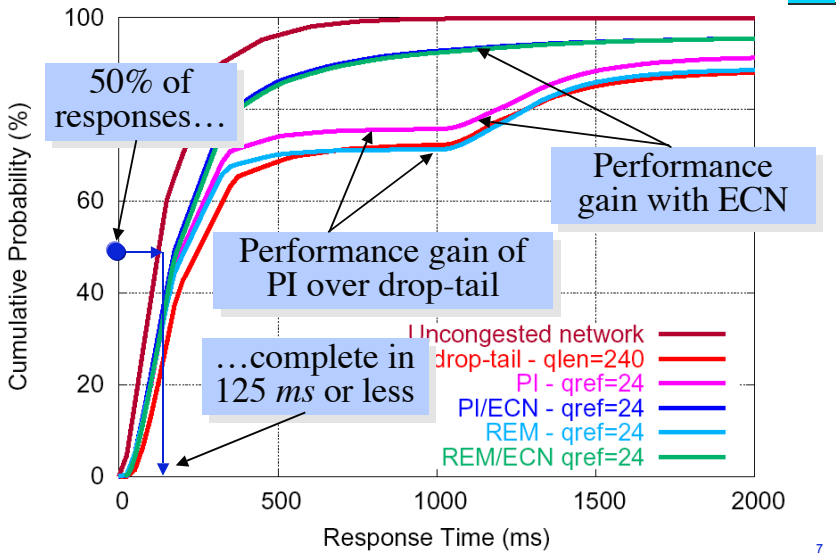


- Set a bit in a packet's header and forward towards the ultimate destination
- A receiver recognizes the marked packet and sets a corresponding bit in the next outgoing ACK
- When a sender receives an ACK with ECN it invokes a response similar to that for packet loss.

Do AQM Schemes Work?
Evaluation of ARED, PI, and REM

- "The Effects of Active Queue Management on Web Performance" [SIGCOMM 2003]. When user response times are important performance metrics:
  - Without ECN, PI results in a modest performance improvement over drop-tail and other AQM schemes
  - With ECN, both PI and REM provide significant performance improvement over drop-tail

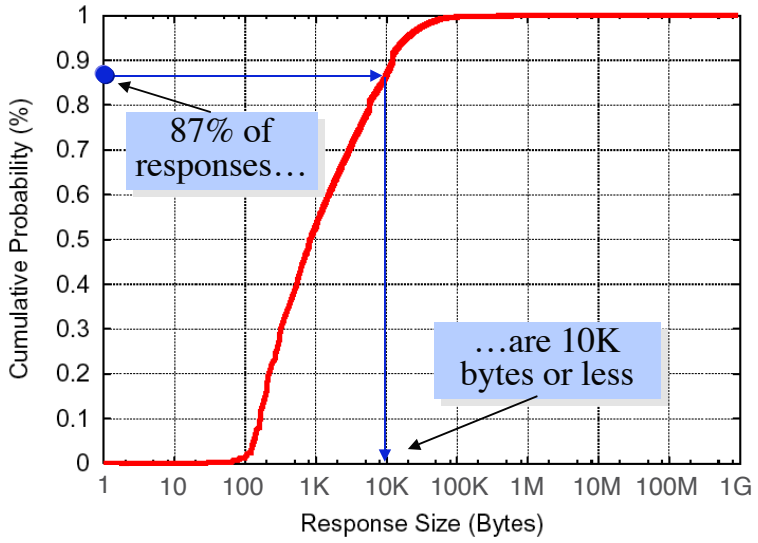
Evaluation of ARED, PI, and REM
Experimental Results – 98% Load



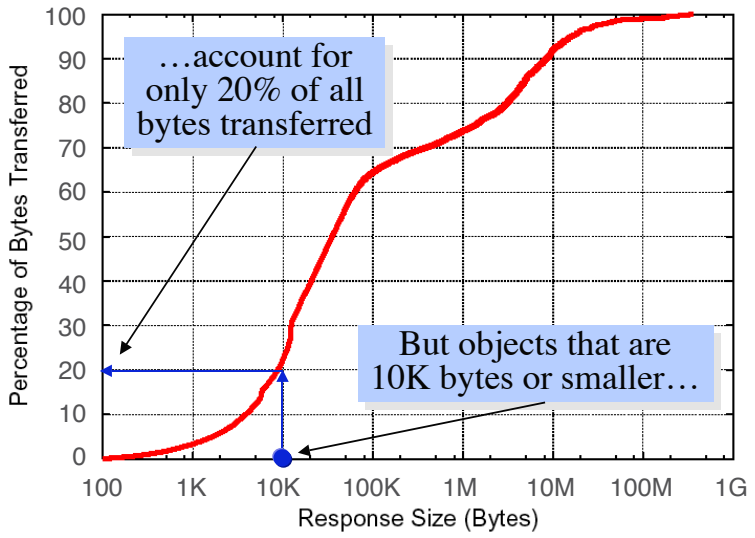
Outline

- Background: Router-based congestion control
  - Active Queue Management
  - Explicit Congestion Notification
- Do AQM schemes work?
- Analysis of AQM performance
  - The case for *differential congestion notification* (DCN)
- A DCN prototype and its empirical evaluation

The Structure of Web Traffic
Distribution of response sizes



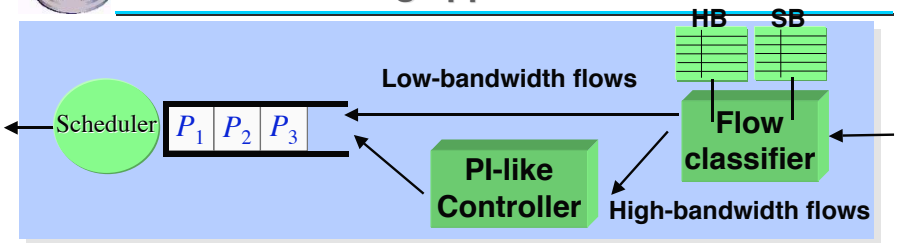
The Structure of Web Traffic
Percent of bytes transferred by response sizes



Realizing Differential Notification
Issues and approach

- How to identify packets belonging to long-lived, high bandwidth flows with minimal state?
  - Adopt the Estan & Varghese flow filtering scheme developed for traffic accounting [SIGCOMM 2002]
- How to determine when to signal congestion (by dropping packets)?
  - Use a PI-like scheme [Infocom 2001]
- Differential treatment of flows an old idea:
  - FRED
  - CHOKe
  - AFD
  - RIO-PS
  - SRED
  - SFB
  - RED-PD
  - …

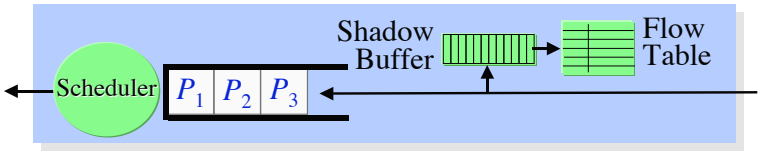
Classifying Flows
A score-boarding approach



- Use two hash tables (hash keys are formed by IP addressing 4-tuple plus protocol number):
  - A "suspect" flow table HB ("high-bandwidth") and
  - A per-flow packet count table SB ("scoreboard")
- Arriving packets from flows in HB are subject to dropping
- Arriving packets from other flows are inserted into SB and tested to determine if the flow should be considered high-bandwidth
  - Use a simple packet count threshold for this determination

An Alternate Approach

AFD [Pan *et al.* 2003]

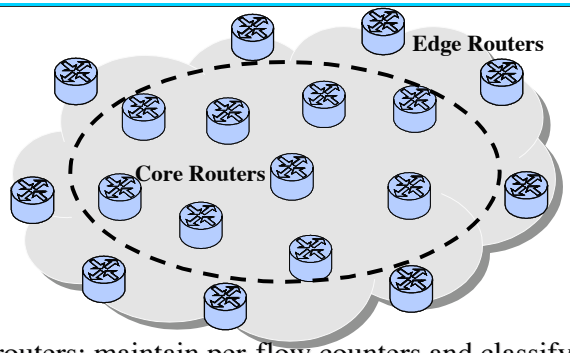


"Approximate Fairness through Differential Dropping"

- Sample 1 out of every *s* packets and store in a *shadow buffer* of size *b*
- Estimate flow's rate as $r_{est} = R \frac{\#\ matches}{b}$
- Drop packet with probability $p = 1 - \frac{r_{fair}}{r_{est}}$

Another Alternate Approach

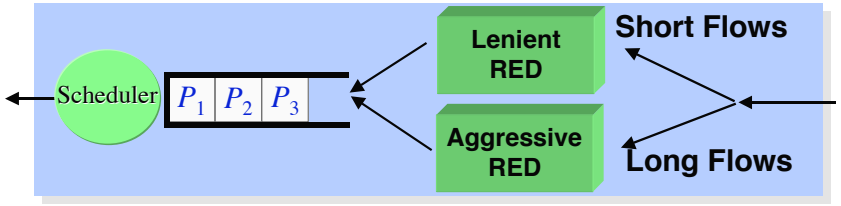
RIO-PS [Guo and Matta 2001]



- Edge routers: maintain per-flow counters and classify flows into two classes: "Short" or "Long"
- Core routers:
  - use different RED engines for short and long flows
  - use different RED parameter settings to give preferential treatment to short flows

Another Alternate Approach

RIO-PS [Guo and Matta 2001]



Core router's architecture

- Edge routers: maintain per-flow counters and classify flows into two classes: "Short" or "Long"
- Core routers:
  - use different RED engines for short and long flows
  - use different RED parameter settings to give preferential treatment to short flows

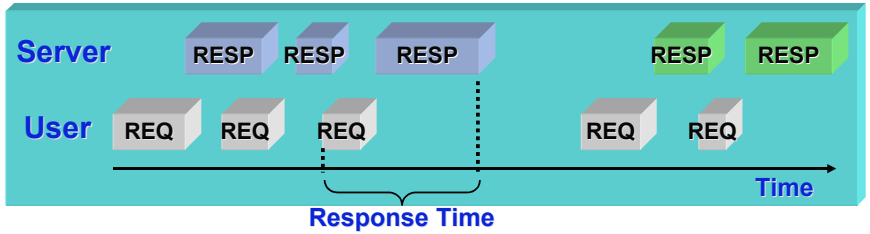
Outline

- Background: Router-based congestion control
  - Active Queue Management
  - Explicit Congestion Notification
- Do AQM schemes work?
- Analysis of AQM performance
  - The case for *differential congestion notification* (DCN)
- A DCN prototype and its empirical evaluation

Evaluation Methodology

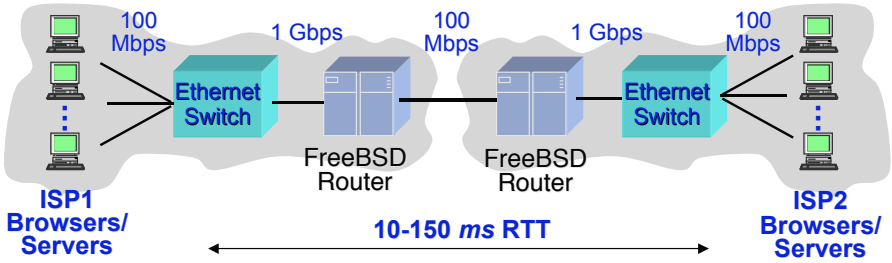
- Evaluate AQM schemes through "live simulation"
- Emulate the browsing behavior of a large population of users surfing the web in a laboratory testbed
  - Construct a physical network emulating a congested peering link between two ISPs
  - Generate synthetic HTTP requests and responses but transmit over real TCP/IP stacks, network links, and switches
  - Also perform experiments with mix of TCP applications

Experimental Methodology
HTTP traffic generation



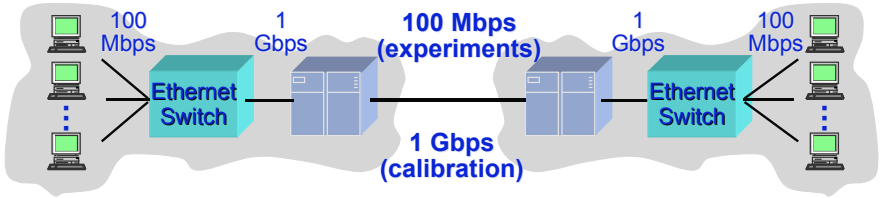
- Synthetic web traffic generated using the UNC HTTP model [SIGMETRICS 2001, MASCOTS 2003]
- Primary random variables:
  - Request sizes/Reply sizes
  - User think time
  - Persistent connection usage
  - Nbr of objects per persistent connection
  - Number of embedded images/page
  - Number of parallel connections
  - Consecutive documents per server
  - Number of servers per page

Experimental Methodology
Testbed emulating an ISP peering link



- AQM schemes implemented in FreeBSD routers using ALTQ kernel extensions
- End-systems either a traffic generation client or server
  - Use *dummynet* to provide *per-flow* propagation delays
  - Two-way traffic generated, equal load generated in each direction

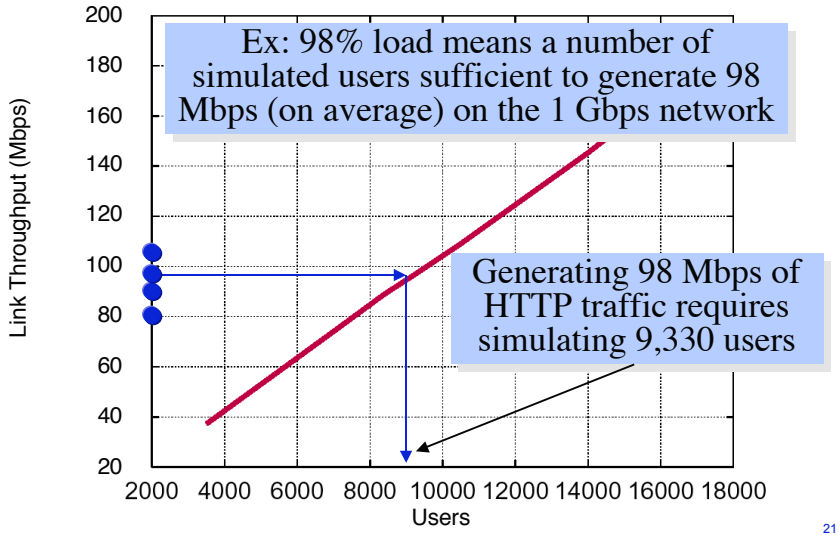
Experimental Methodology
1 Gbps network calibration experiments



- Experiments run on a congested 100 Mbps link
- Primary simulation parameter: Number of simulated browsing users
- Run calibration experiments on an uncongested 1 Gbps link to relate simulated user populations to average link utilization
  - (And to ensure offered load is linear in the number of simulated users — *i.e.*, that end-systems are not a bottleneck)

Experimental Methodology

1 Gbps network calibration experiments



DCN Evaluation

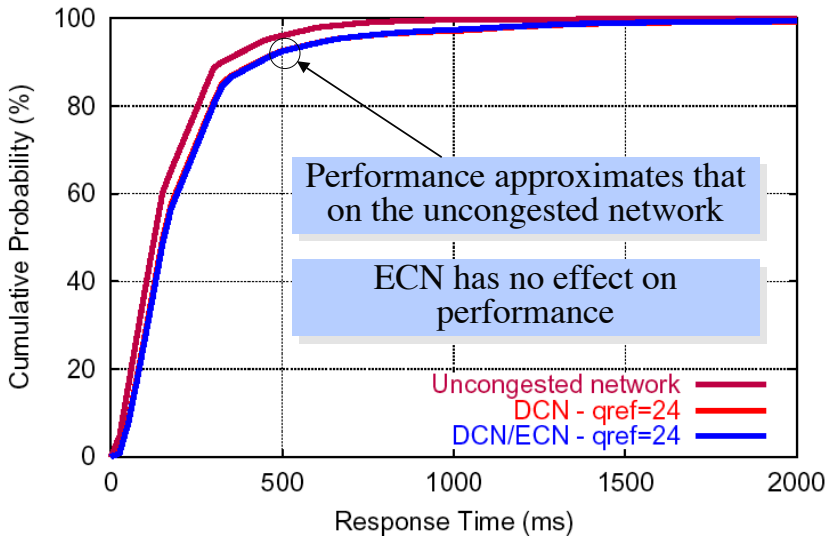
Experimental plan

	80%	90%	98%	105%
uncongested	*loss rate*			
drop-tail	*utilization*			
DCN	*response times*			
AFD	*completed requests*			
PI				
RIO-PS				

- Run experiments with DCN, AFD, RIO-PS, and PI at different offered loads
  - PI always uses ECN, test AFD and RIO-PS with and without ECN
  - DCN always signals congestion via drops
- Compare DCN results against…
  - The better of PI, AFD, and RIO-PS (the performance to beat)
  - The uncongested network (the performance to approximate)

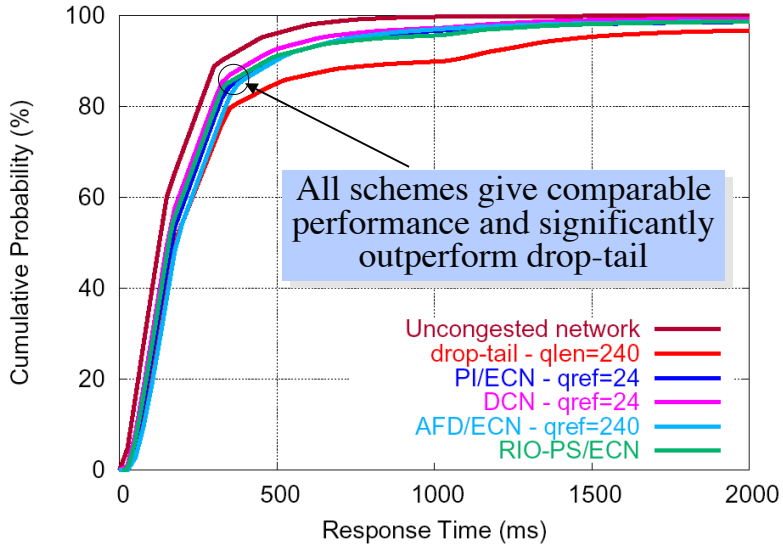
Experimental Results — 90% Load

DCN performance

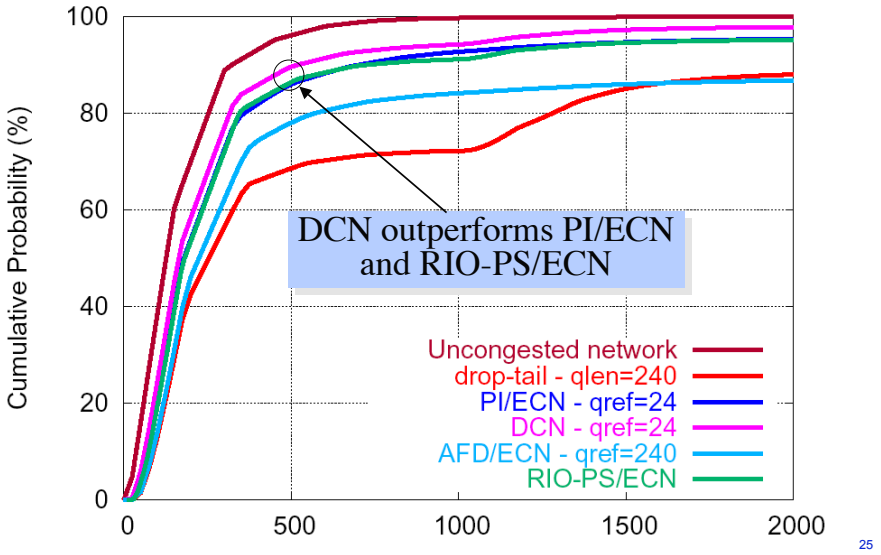


Experimental Results — 90% Load

Comparison of all schemes



Experimental Results — 98% Load
Comparison of all schemes



DCN Evaluation
Summary

- DCN uses a simple, tunable two-tiered classification scheme with:
  - Tunable storage overhead
  - $O(1)$ complexity with high probability
- DCN, without ECN, meets or exceeds the performance of the best performing AQM designs with ECN
  - The performance of 99+% of flows is improved
  - More small and "medium" flows complete per unit time
- On heavily congested networks, DCN closely approximates the performance achieved on an uncongested network

Summary and Conclusions

- For offered loads of 90% or greater there is benefit to control theoretic AQM but only when used with ECN
- Heuristically signaling only long-lived, high-bandwidth flows improves the performance of most flows and eliminates the requirement for ECN
  - One can operate links carrying HTTP traffic at near saturation levels with performance approaching that achieved on an uncongested network
- Identification of high-bandwidth flows can be effectively performed with tunable overhead and complexity

The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Differential Congestion Notification: Taming the Elephants

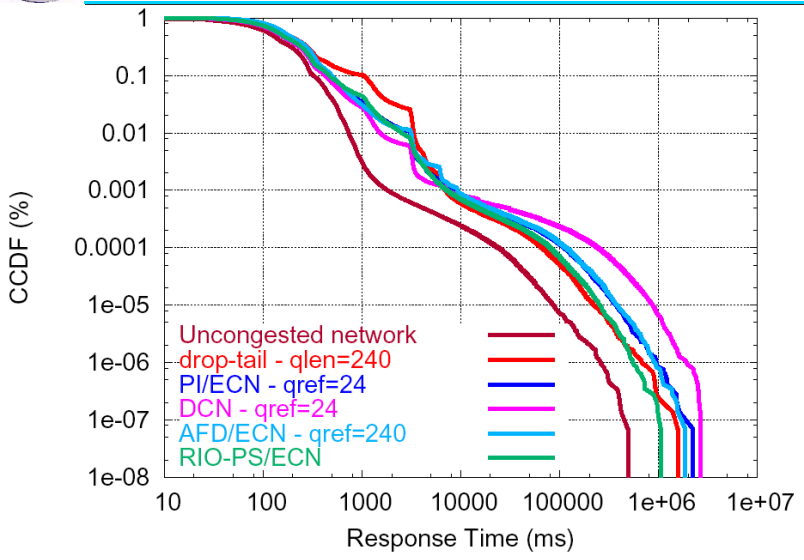
Long Le, Jay Aikat, Kevin Jeffay, and Don Smith

IEEE ICNP 2004

http://www.cs.unc.edu/Research/dirt

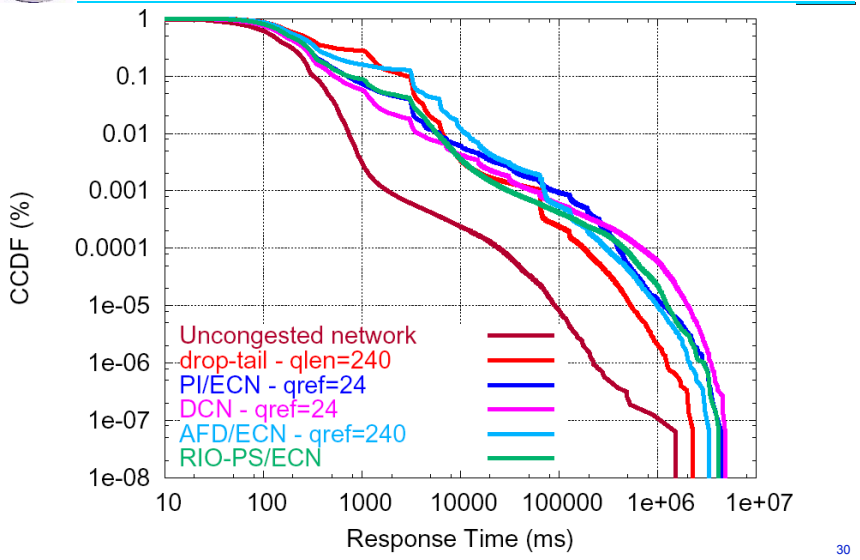
Experimental Results — 90% Load

Comparison of all schemes (CCDF)



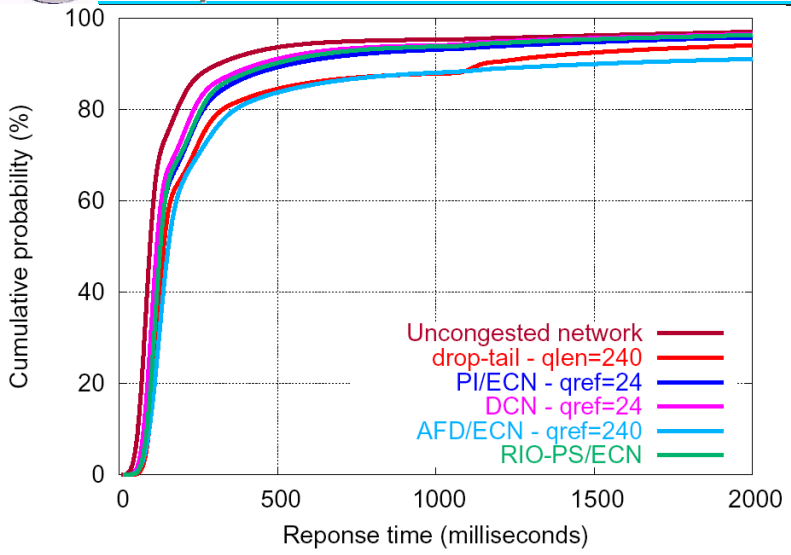
Experimental Results — 98% Load

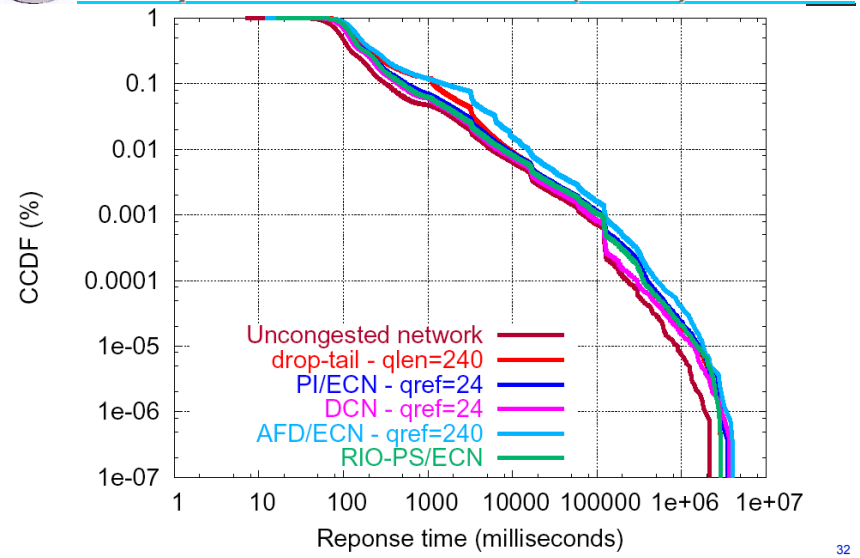
Comparison of all schemes (CCDF)



Experimental Results with General TCP Traffic

Comparison of all schemes



Experimental Results with General TCP Traffic

Comparison of all schemes (CCDF)