Tuning RED for Web Traffic

Mikkel Christiansen, Kevin Jeffay,
David Ott, Donelson Smith

UNC at Chapel Hill

SIGCOMM 2000

Stockholm

Tuning RED Outline

- Introduction
- Background and Related Work
- Experimental Methodology
  - Web-like Traffic Generation
- Experiment Calibrations and Procedures
- FIFO and RED Results
- Conclusions

Introduction

- RFC2309 recommends Active Queue Management [AQM] for Internet congestion avoidance.
- RED, the best known AQM technique, has not been studied much for **Web traffic**, the dominant subset of TCP connections on the Internet today.
- The authors use **response time**, a user-centric performance metric, to study short-lived TCP connections that model HTTP 1.0.

Introduction

- They model HTTP request-response pairs in a lab environment that simulates a large collection of *browsing* users.
- Artificial delays are added to a small lab testbed to approximate coast-to-coast US round trip times (RTT's).
- The paper focuses on studying RED tuning parameters.
- The basis of comparison is the effect of RED vs. Drop Tail on response time for HTTP 1.0.

Background and Related Work

- Authors review RED parameters (*avg*, *qlen*, min_{th}, max_{th}, w_q , max_p) and point to Sally Floyd guidelines.
- RED is effective in preventing congestion collapse when TCP windows configured to exceed network storage capacity.
- Claim:: The bottleneck router queue size should be 1-2 times the bandwidth-delay product.
- RED issues (shortcomings) studied through alternatives: BLUE, Adaptive RED, BRED, FRED, SRED, and Cisco's WRED.

Background and Related Work

- ECN not considered in this paper.
- Big deal:: most of the previous studies used small number of sources except BLUE paper with 1000-4000 Parento on-off sources (but BLUE uses ECN).
- Previous tuning results include:
  - max_p is dependent on the number of flows
  - router queue length stabilizes around max_{th} for a large number of flows

Background and Related Work

- Previous analytic modeling at INRIA results:
  - **TCP goodput** **does not** improve significantly with RED and this effect is independent of the number of flows.
  - RED has lower mean queueing delay but higher variance.
- Conclusion – research pieces missing include: Web-like traffic and worst-case studies where there are dynamically changing number of TCP flows with highly variable lifetimes.

Experimental Methodology

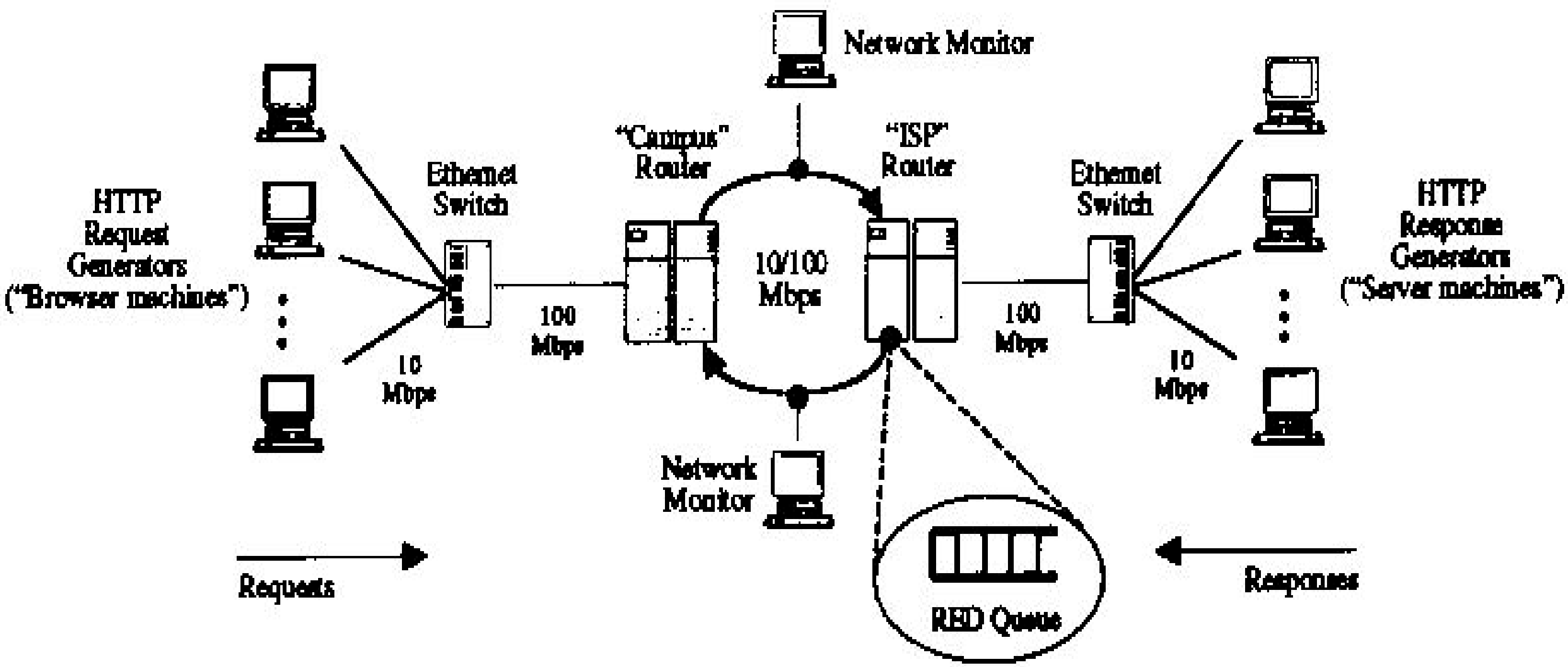

Figure 2: Experimental laboratory network diagram.

Experimental Methodology

These researchers used careful, meticulous, experimental techniques that are excellent.

- They use FreeBSD 2.2.8, ALTQ version 1.2 extensions, and *dummynet* to build a lab configuration that emulates full-duplex Web traffic through two routers separating Web request generators {*browser machines*} from Web servers.
- They emulate RTT's uniformly selected from 7-137 ms. range derived from measured data.
- FreeBSD default TCP window size of 16KB was used.

Experimental Methodology

- Monitoring tools:
  - At router interface collect: router queue size mean and variance, max queue size, min queue size sampled every 3 ms.
  - The machine connected to hubs forming links to routers uses a modified version of *tcpdump* to produce log of link throughput.
  - end-to-end measurements done on end-systems (e.g., response times)

Web-like Traffic Generation

- The traffic for the experiments is based on Mah's Web browsing model that include:
  - HTTP request length in bytes
  - HTTP reply length in bytes
  - The number of embedded (file) references per page
  - The time between retrieval of two successive pages (user think time)
  - The number of consecutive pages requested from a server.

Web-like Traffic Generation

- The empirical distributions for all these elements were used in synthetic-traffic generators built.
- The client-side request-generation program emulates behavioral elements of Web browsing.
- important parameters: number of browser users (several hundred!!) the program represents and the user think time
- A new TCP connection is made for each request/response pair.
- Another parameter: number of concurrent TCP connections per browser user.

Experiment Calibrations and Procedures

1. They needed to insure that congested link between routers was the **primary bottleneck** on the end-to-end path.
2. They needed to guarantee that the offered load on the testbed network could be predictably controlled using the **number of emulated browser users** as a parameter to the traffic generator.

Experiment Calibrations and Procedures

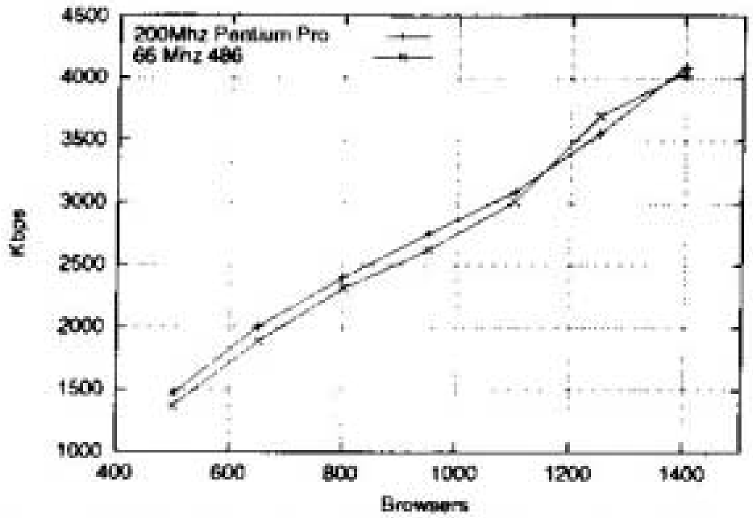

Figure 3: Offered load as a function of the number of simulated users on one machine.

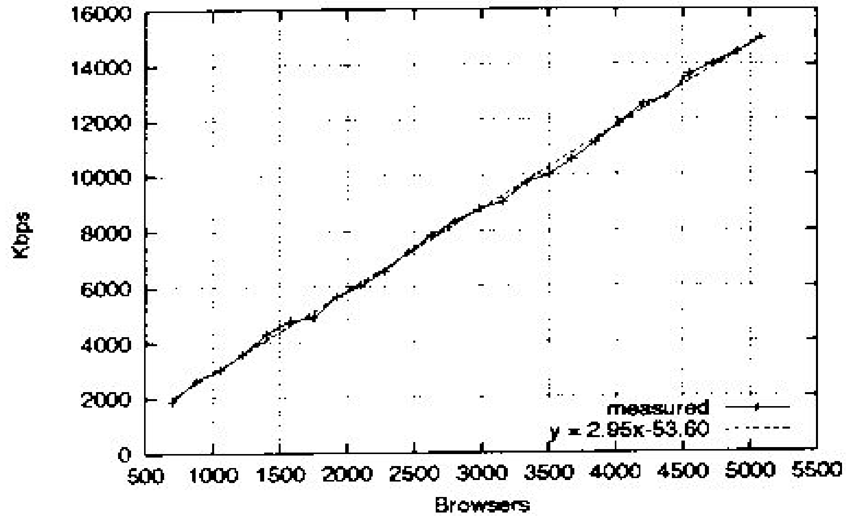

Figure 4: Offered load as a function of the number of simulated users on 7 machines.

Figure 3 and 4 show desired linear increases that imply no fundamental resource limitations.

The authors were concerned about exceeding 64 socket descriptors limitation on one FreeBSD process. This limit was never encountered due to long user think times.

Experiment Calibrations and Procedures

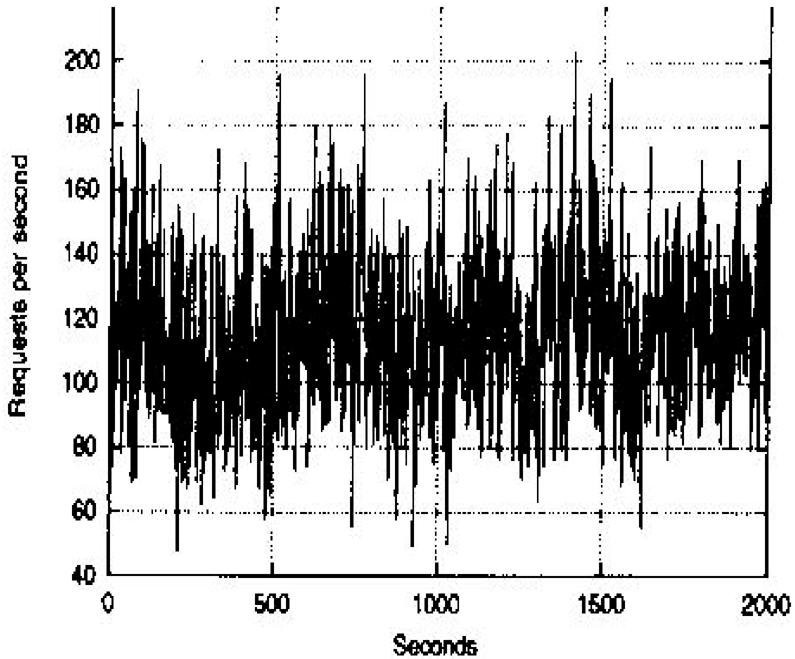

Figure 5: Requests per second from 3,500 users.

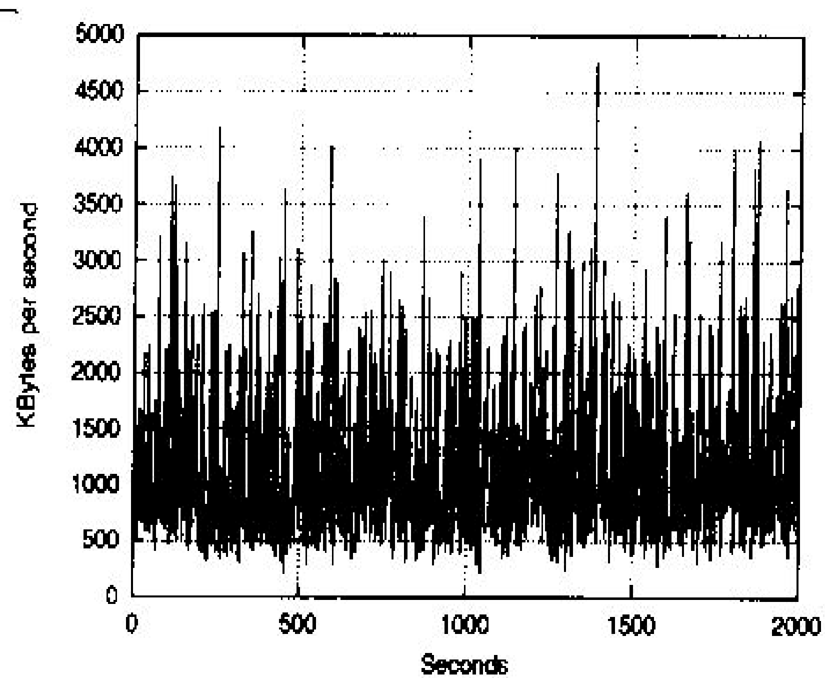

Figure 6: Bytes requested per second from 3,500 users.

Figures 5 and 6 show highly bursty nature of requests by 3500 users during one second intervals.

Experimental Procedures

- After initializing and configuring the testbed, the server-side processes were started followed by the browser processes.
- Each browser emulated an equal number of users chosen to place load on network that represent 50, 70, 80, 90, 98 or 110 percent of 10 Mbps capacity.
- Each experiments ran for 90 minutes with the first 20 minutes discarded to eliminate startup and stabilization effects.

Experiment Calibrations and Procedures

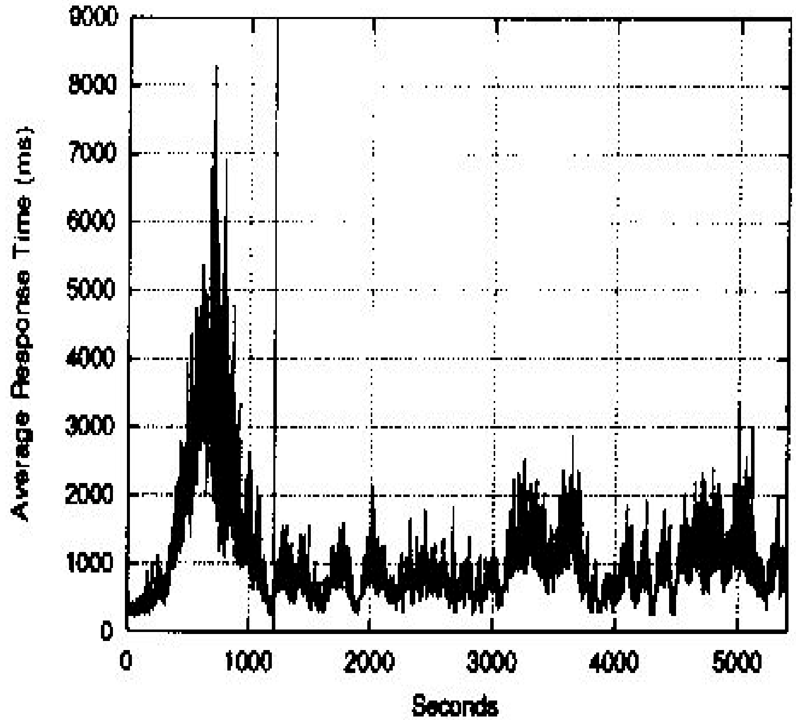

Figure 7: Average response time per second during an experiment. The plot includes the initial 20 minutes, where the traffic generators are started and stabilize.

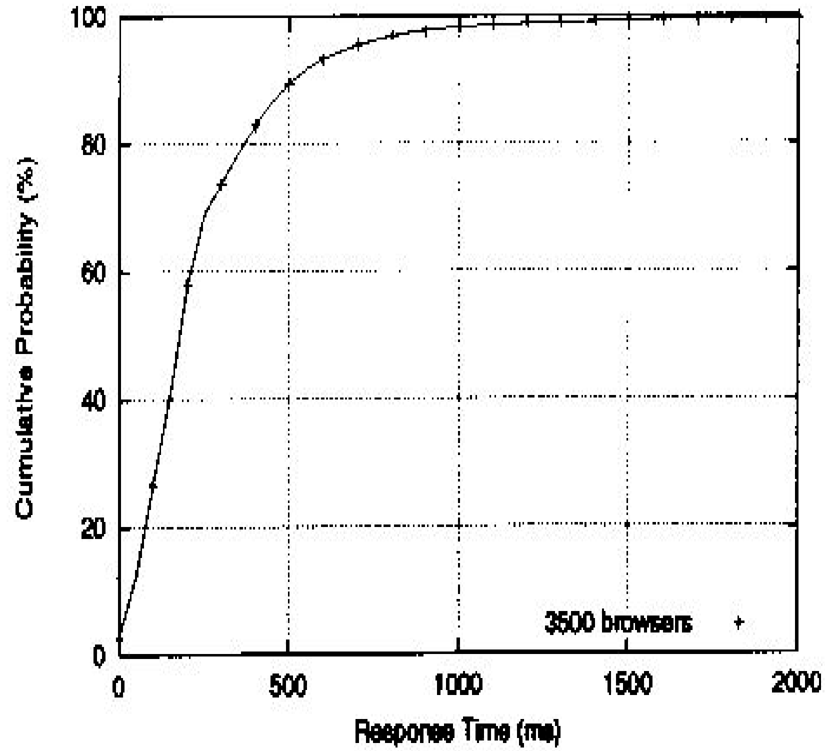

Figure 8: Cumulative response time distribution for 3,500 users on the unconstrained (100 Mbps) network.

Experimental Procedures

- Figure 8 represents the ***best-case performance*** for 3500 browsers generating request/response pairs in an unconstrained network.
- Since responses from the servers are much larger than requests to server, *only* effects on IP output queue carrying traffic from servers to browsers is reported.
- They measure: end-to-end response times, percent of IP packets dropped at the bottlenecked link, mean queue size and throughput achieved on the link.

FIFO Results [Drop Tail]

- FIFO tests run to establish a baseline.

* For the critical FIFO parameter, *queue size*, the consensus is roughly 2-4 times the *bandwidth-delay product* (bdp)

  – mean min RTT = 79 ms.

  \+ 10 Mbps congested link => 96 K bytes (bdp)

  – measured IP datagrams approx. 1 K bytes ➔ 190 - 380 elements in FIFO queue!

FIFO Results

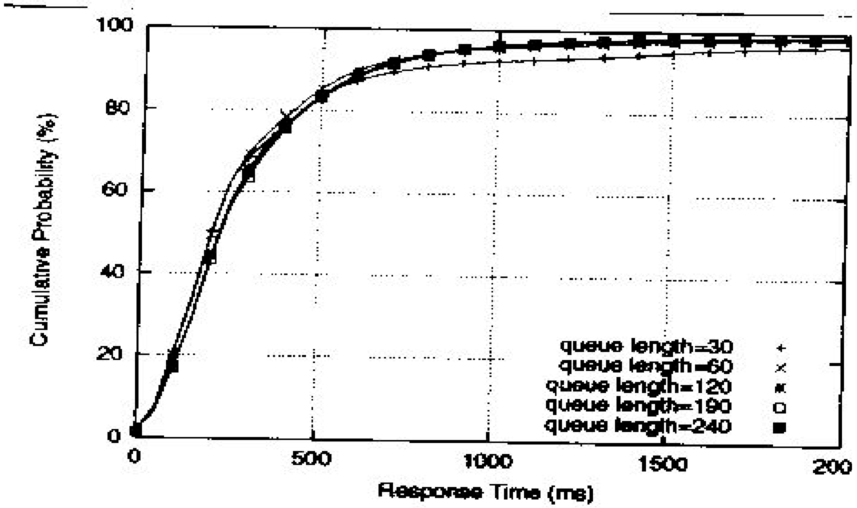

Figure 9a: FIFO performance at 80% load.

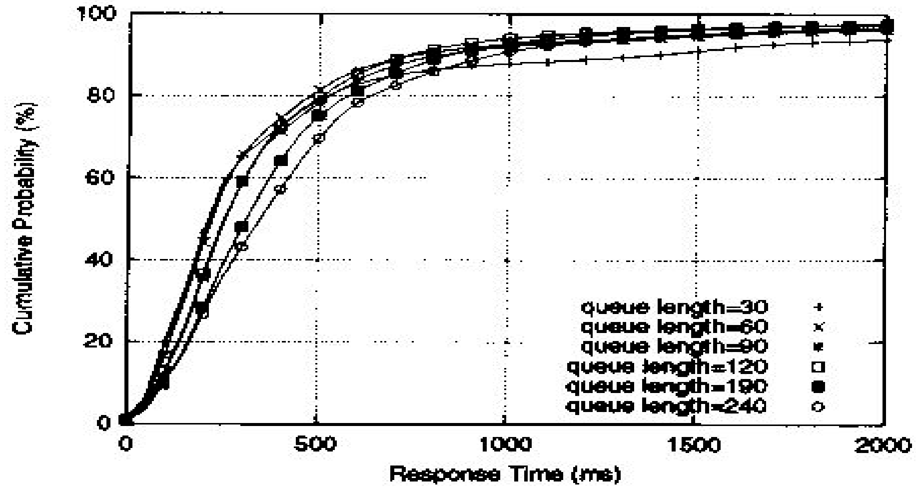

Figure 9b: FIFO performance at 90% load.

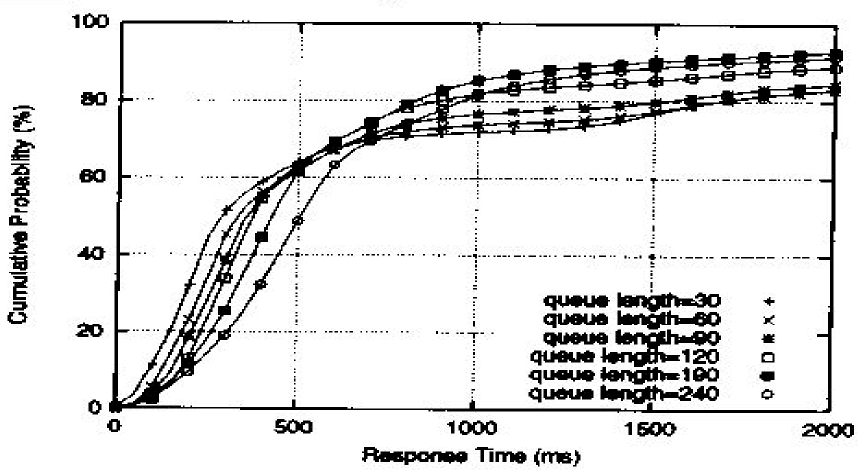

Figure 9c: FIFO performance at 98% load.

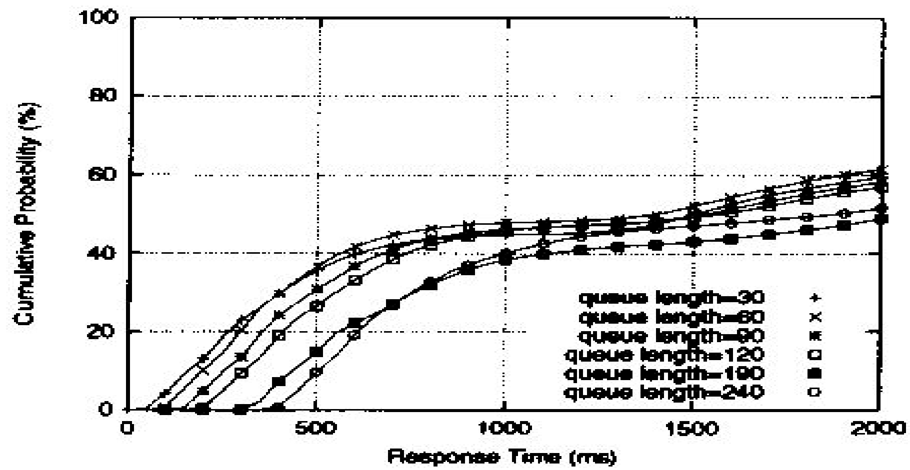

Figure 9d: FIFO performance at 110% load.

Figure 9 FIFO Results

- A queue size of from 120 to 190 is a reasonable choice especially when one considers the tradeoffs for response time without significant loss in link utilization or high drops.
- At 98% (Figure 9c), the

FIFO Results

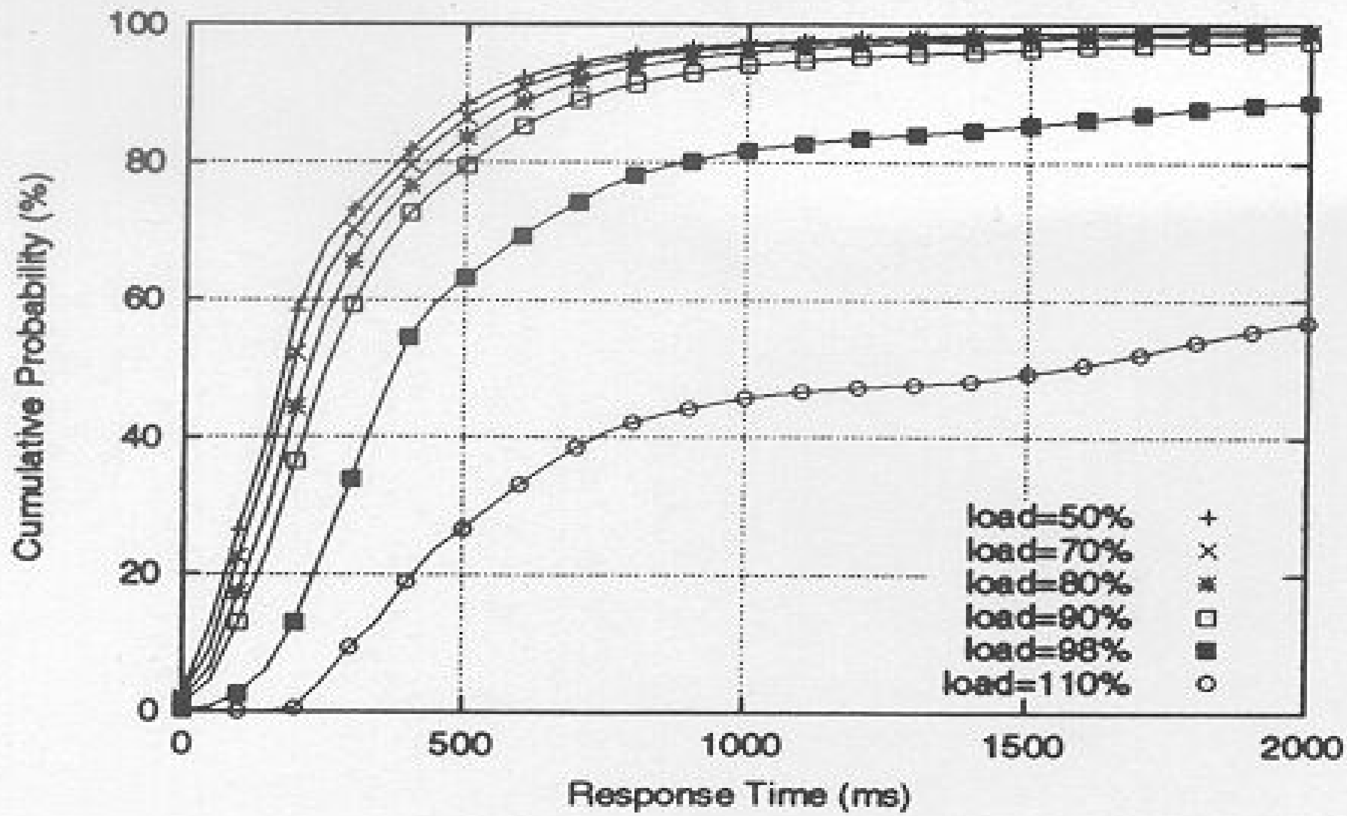

Figure 10: FIFO performance for different loads with a queue length of 120 elements.

FIFO Results

From Figure 10

- At loads below 80% capacity, there is no significant change in response time as a function of load.
- Response time degrades sharply when offered load exceeds link capacity.

RED Results

- Experimental goal: determine parameter settings that provide good performance for RED with Web-traffic.
- Another objective is to examine the tradeoffs in RED tuning parameter choices
- The FIFO results show complex tradeoff between response times for short responses and response times for longer responses.

RED Results

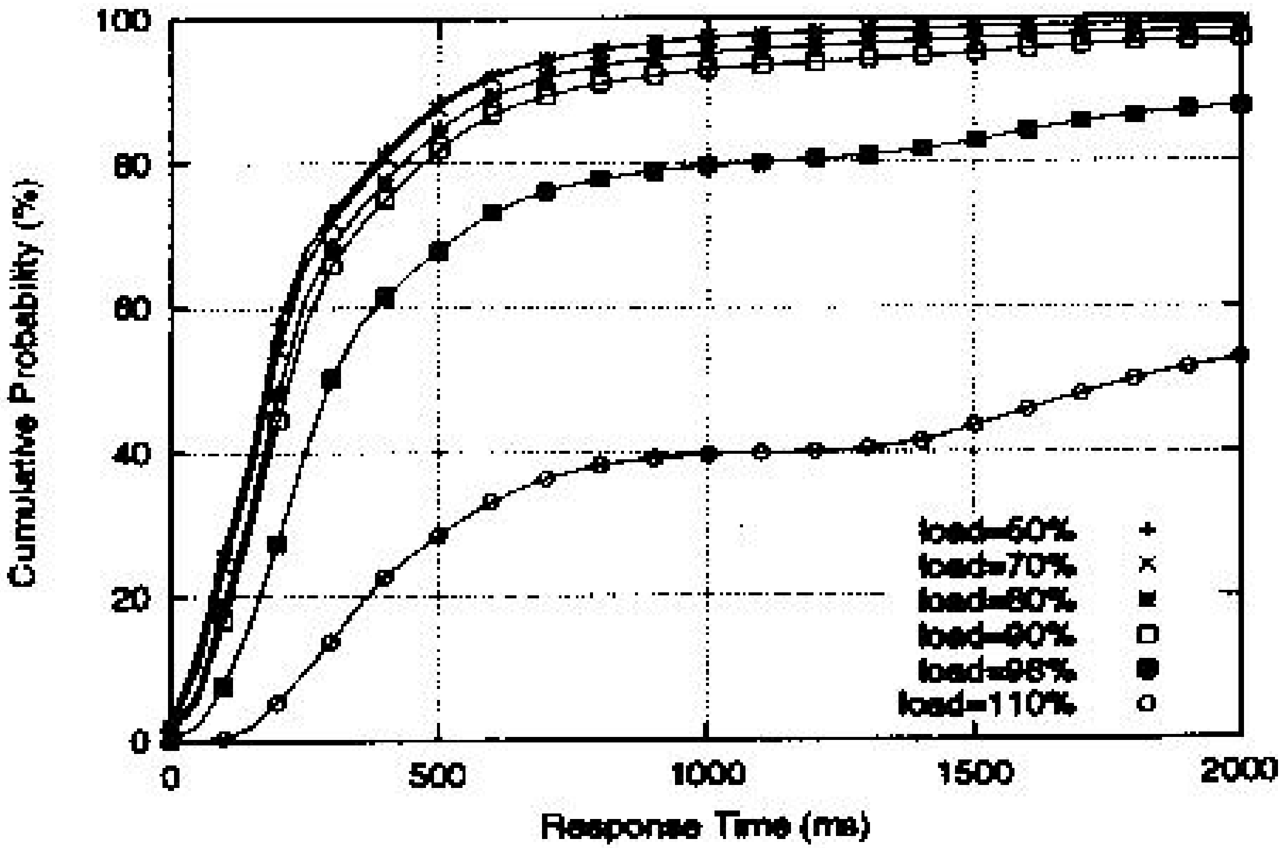

Figure 11: The performance of RED at different loads. $w_q = 1/512$, $max_p = 1/10$, $min_{th} = 30$, $max_{th} = 90$, $qlen = 480$.

RED Results

{set queue size to 480 to eliminate physical queue length (*qlen*) as a factor}

Figure 11: shows the effect of varying loads on response time distributions.

- (min_{th} , max_{th}) set to (30, 90)
- The interesting range for varying RED parameters for optimization is between 90-110% load levels where performance decreases significantly.

RED Results

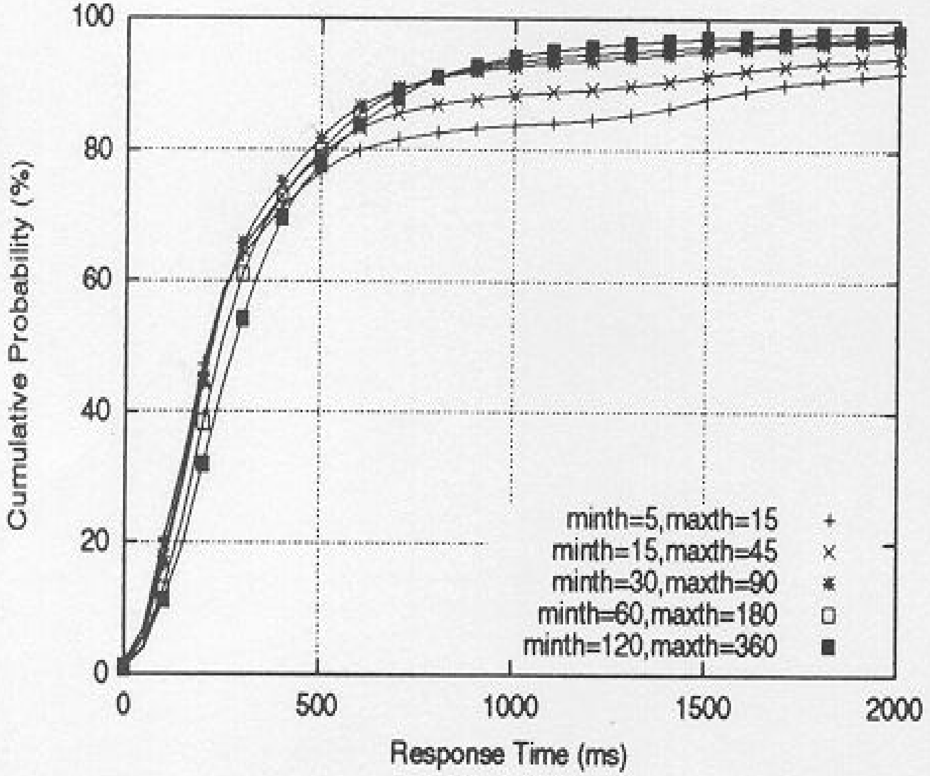

Figure 12a: Response time CDF for offered load at 90% of link capacity (w_q=1/512, max_p=1/10, *qlen*=480).

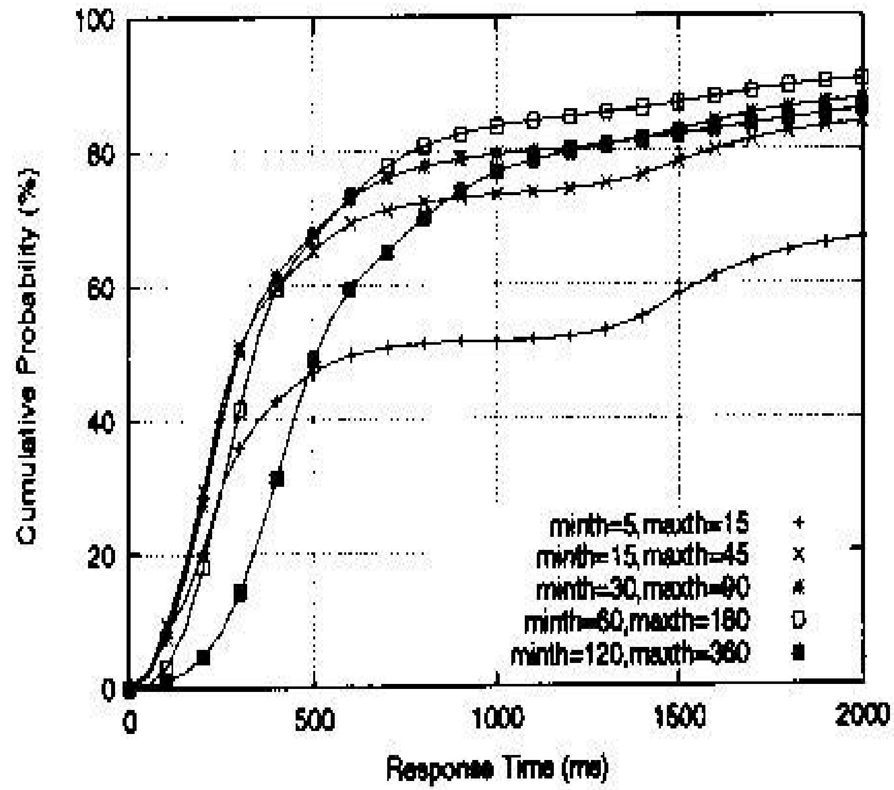

Figure 12b: Response time CDF for offered load at 98% of link capacity (w_q=1/512, max_p=1/10, *qlen*=480).

RED Results

Figure 12 {load at 90% and 98%}

- study min_{th} , max_{th} choices
  - Floyd choice (5, 15) => poor performance
- (30, 90) or (60, 180) are best choices!

RED Results

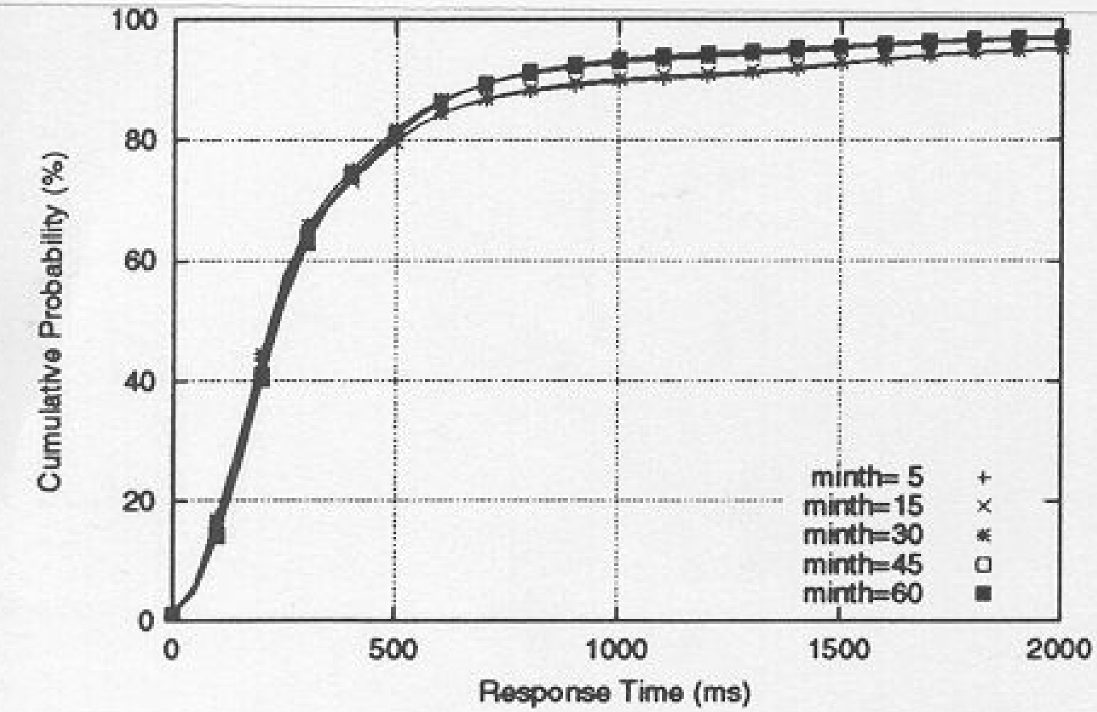

Figure 13: The effect of changing min_{th}. Load = 90% and max_{th} = 90, w_q = 1/512, max_p = 1/10, $qlen$ = 480.

The effect of varying min_{th} is small at 90% load.

RED Results

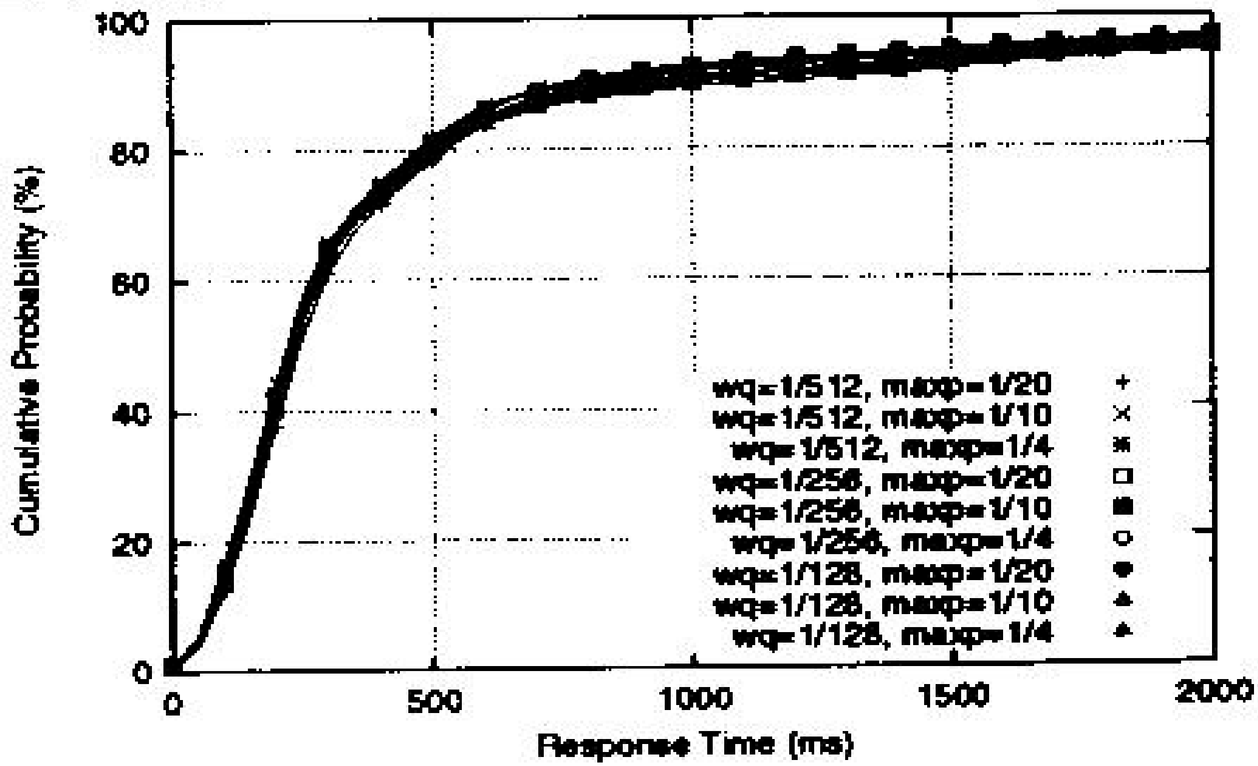

Figure 14: Results for different values of w_q and max_p. Load = 90%, and $qlen$ = 480, min_{th} = 30, max_{th} = 90.

Figure 14 RED Results

- max_p = 0.25 has negative impact on performance – too many packets are dropped. Generally, changes in w_q and $\max_p$ mainly impact longer flows

Table 3 Limiting Queue Size

- 120 good choice for queue size

* **only min_{th}** setting needs to be changed due to bursty network traffic.

RED Results

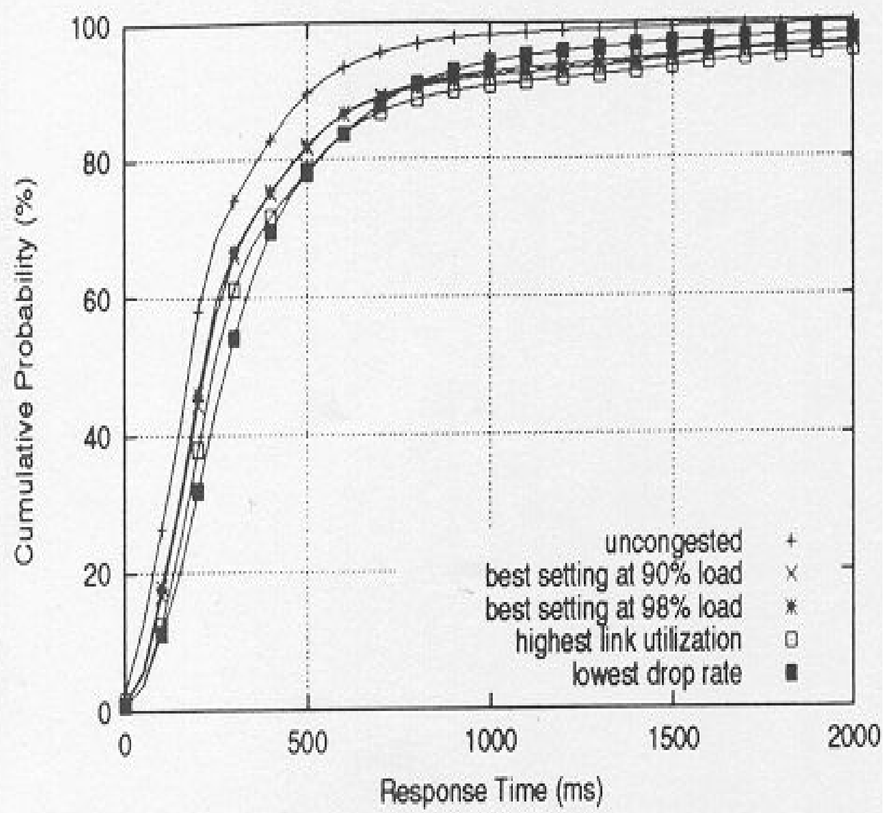

Figure 15a: "Good" RED parameter settings at 90% load.

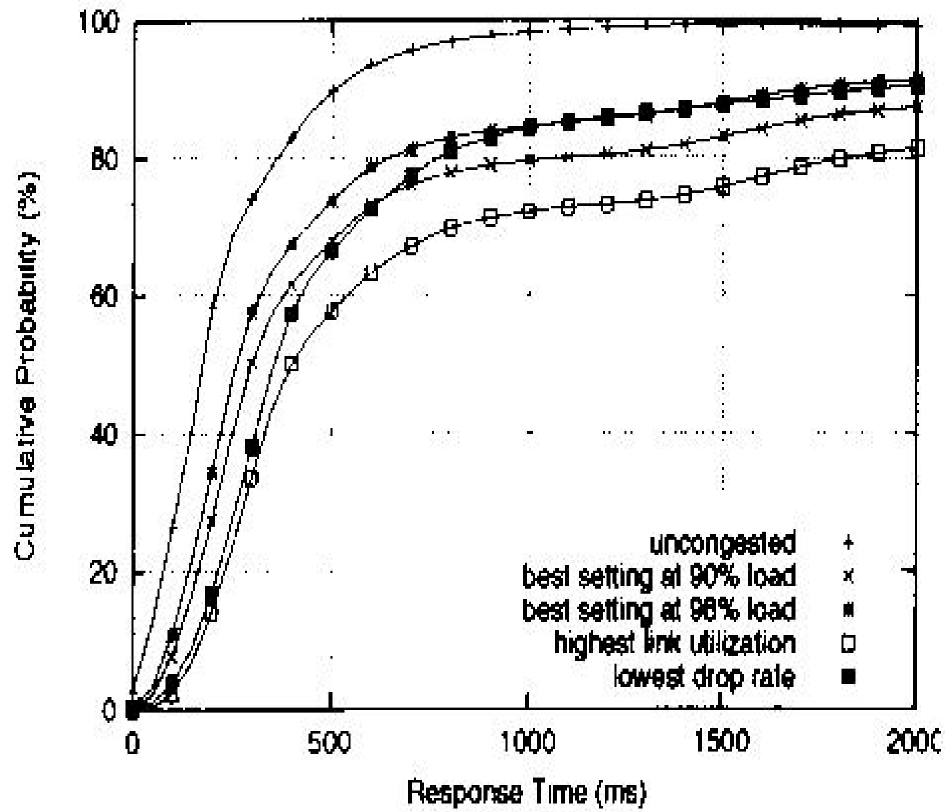

Figure 15b: "Good" RED parameters settings at 98% load.

RED Results

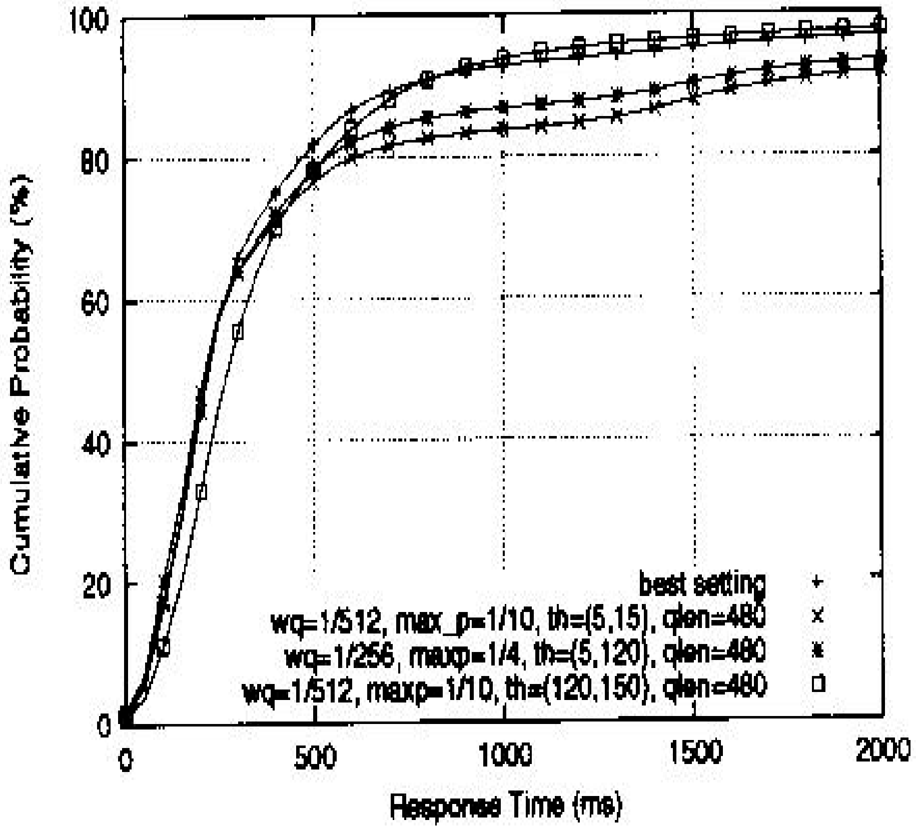

Figure 16a: "Bad" RED parameters settings at 90% load.

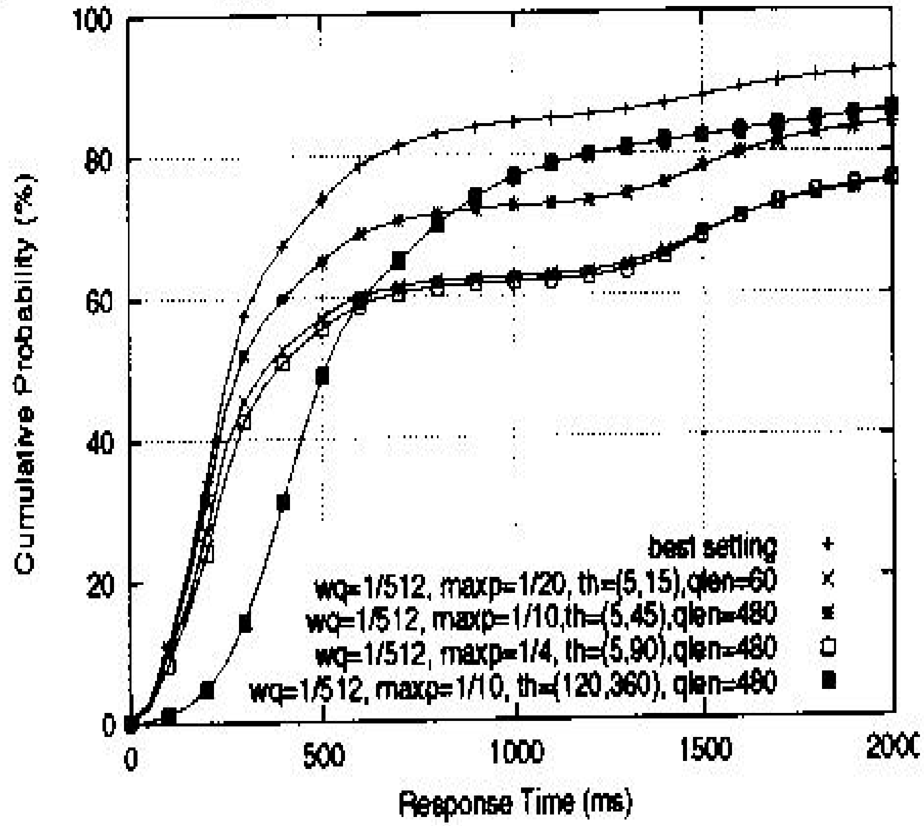

Figure 16b: "Bad" RED parameters settings at 98% load.

Figures 15 and 16 RED Results

- RED can be tuned to yield "best settings" for a given load percentage
- at high loads, near saturation, there is a significant downside potential for choosing "bad" parameter settings

bottom line: tuning is not easy!

Analysis of RED Response Times

- New section added
- Detailed analysis of retransmission patterns for various TCP segments (e.g., SYN, FIN)
- This section reinforces the complexity of understanding the effects of RED for HTTP traffic.

RED Results

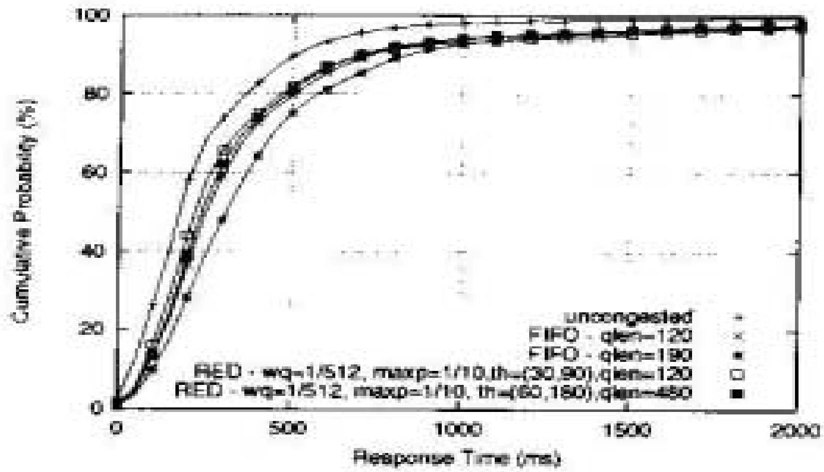

Figure 22a: FIFO and RED at 90% load.

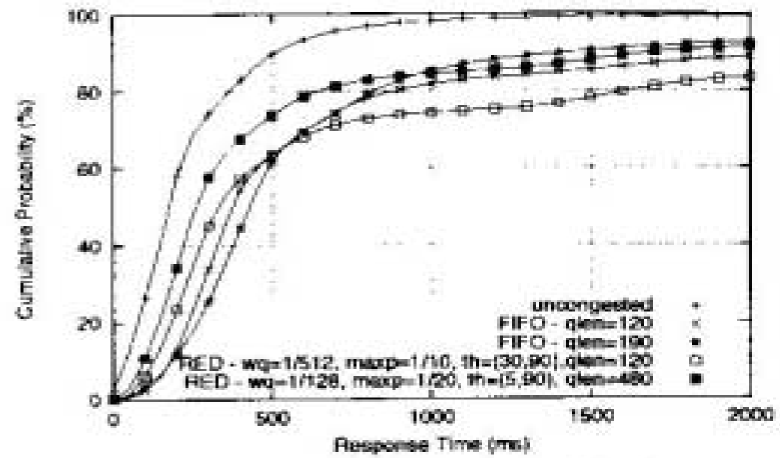

Figure 22b: FIFO and RED at 98% load.

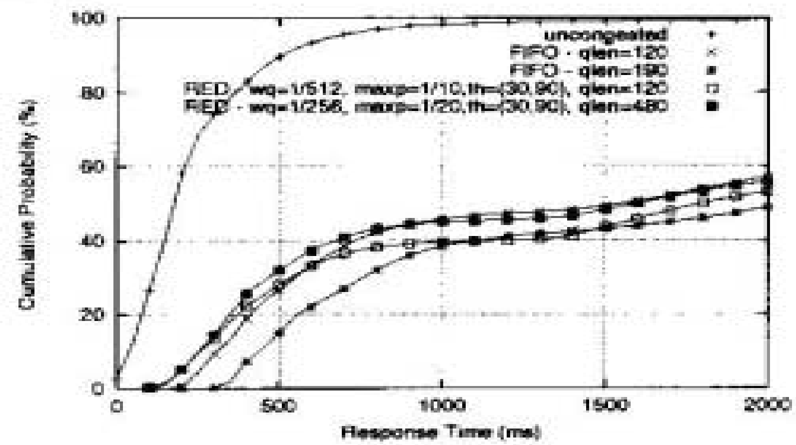

Figure 22c: FIFO and RED at 110% load.

Figure 22 FIFO vs. RED

The only improvement for RED is at 98% load where careful tuning improves response times for shorter responses.

Conclusions

- Contrary to expectations, there is little improvement in response times for RED for offered loads up to 90%.
- At loads approaching link saturation, RED can be carefully tuned to provide better response times.
- Above 90%, load response times are more sensitive to RED settings with a greater downside potential of choosing bad parameter settings.

* There seems to be no advantage to deploying RED on links carrying only Web traffic.

Question: Why these results for these experiments?