Adaptive RED:

An Algorithm for Increasing the Robustness of RED's Active Queue Management

or

How I learned to stop worrying and love RED

- **Presented by:**
- **Frank Posluszny**
- **Vishal Phirke**

- Introduction
- Background and Related Work
- Metrics and Scenarios
- Pre-results
- Adaptive RED Algorithm.
- Parameters and their values
- Simulations.
- Delay-Throughput Tradeoff.
- Conclusions.

Introduction - 1

- **Who are they authors?**
  - **Sally Floyd (original RED author)**
  - **Ramakrishna Gummadi (CS grad - intern)**
  - **Scott Shenker (works with Sally Floyd)**

Introduction - 2

- **Goals:**
  - **People want a guaranteed delay, which RED can't do without constantly adjusting the parameters**
  - **"Our goal… is to solve this problem with minimal changes to the overall RED algorithm."**

- Introduction
- Background and Related Work
- Metrics and Scenarios
- Pre-results
- Adaptive RED Algorithm.
- Parameters and their values
- Simulations.
- Delay-Throughput Tradeoff.
- Conclusions.

Background & Related Work - 1

- **Quick review of RED**
  - **Try to maintain queue size under a threshold, assuming that as we get closer to that threshold then congestion will start to occur**
  - **Once we "foresee" congestion, drop with an increasing probability**

Background & Related Work - 2

- **Problems, problems everywhere…**
  - **Tuning RED for Web Traffic**
  - **A number of papers point out problems with oscillations in the instantaneous queue size (Misra et at., Hollot et al., Firoiu et al.)**
  - **Average queuing delay**
  - **Throughput**

Background & Related Work - 3

- **Suggested fixes…**
  - **Jacobson (how to set w_q)**
  - **Ziegler (tighter bound for ave_q)**
  - **Feng (adapt max_p)**
  - **AVG (keep queue size small, token bucket)**
  - **SRED (estimate #of active flows)**
  - **DRED (keep queue size near a threshold)**

- Introduction
- Background and Related Work
- Metrics and Scenarios
- Pre-results
- Adaptive RED Algorithm.
- Parameters and their values
- Simulations.
- Delay-Throughput Tradeoff.
- Conclusions.

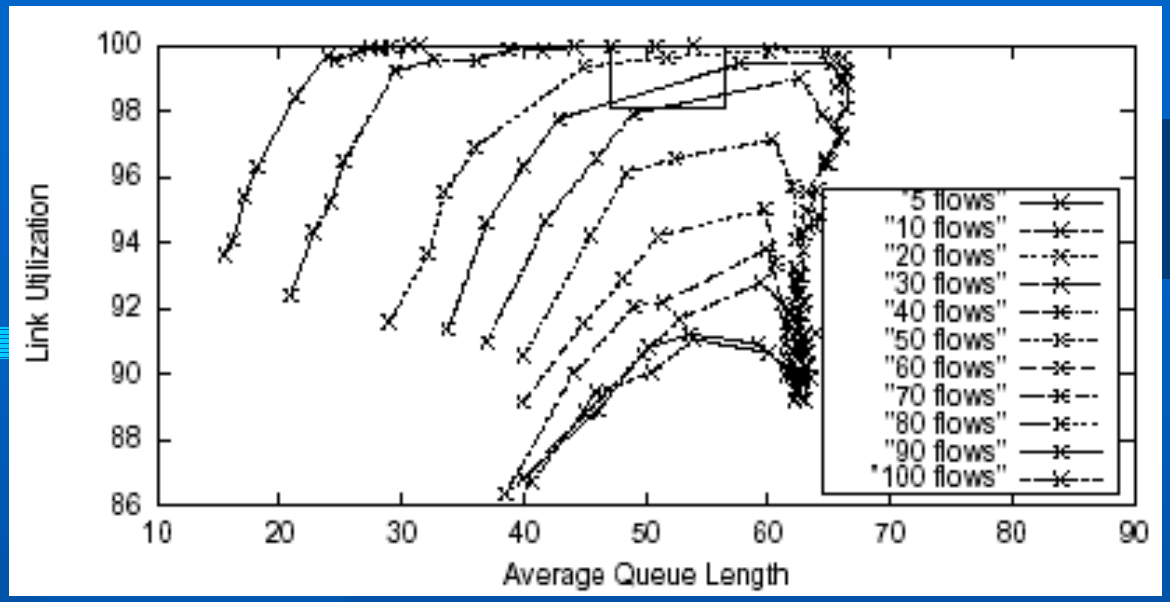
Metrics & Scenarios - 1

- the NS network simulator is used for all tests/scenarios
- router-based metrics vs. user-based
- worst-case is not their concern
- not looking at queue length oscillations directly

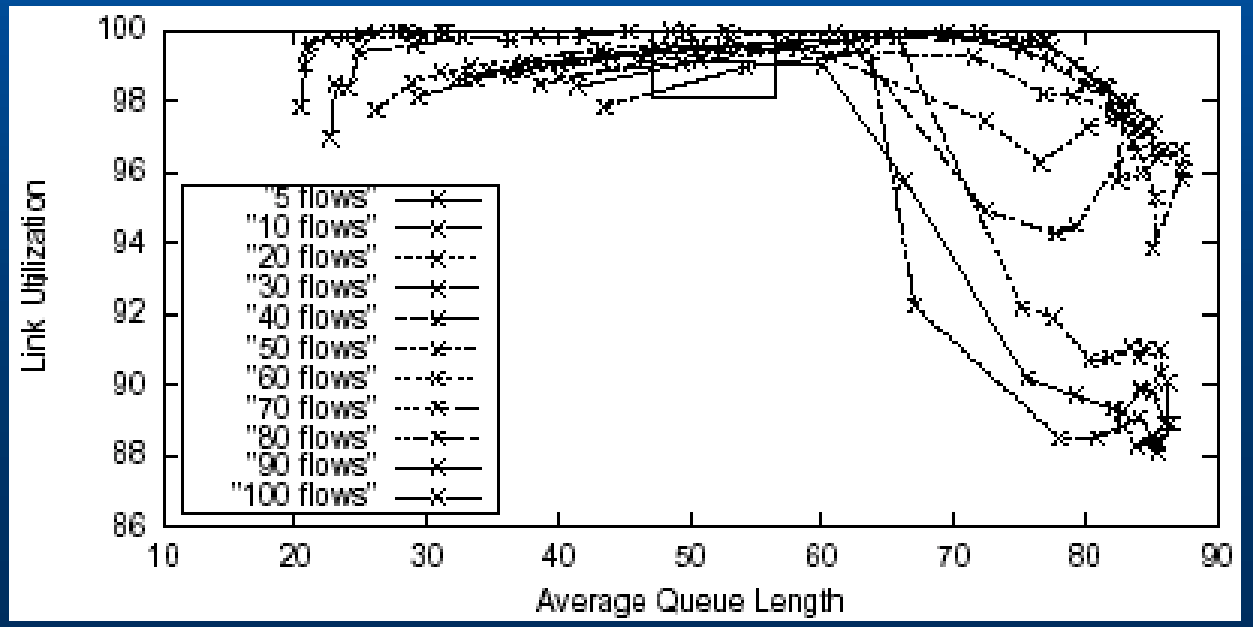
Metrics & Scenarios - 2

- **Wide range or traffic scenarios**
  - **range of workloads (long vs. short lived)**
  - **levels of statistical multiplexing**
  - **levels of congestion**
  - **reverse traffic**
  - **with & without ECN**
  - **large window advertisements**
  - **different packet sizes**

- Introduction
- Background and Related Work
- Metrics and Scenarios
- Pre-results
- Adaptive RED Algorithm.
- Parameters and their values
- Simulations.
- Delay-Throughput Tradeoff.
- Conclusions.



Delay-Utilization tradeoff with RED. w_q = 0.002



Delay-Utilization tradeoff with RED. w_q = 0.00026

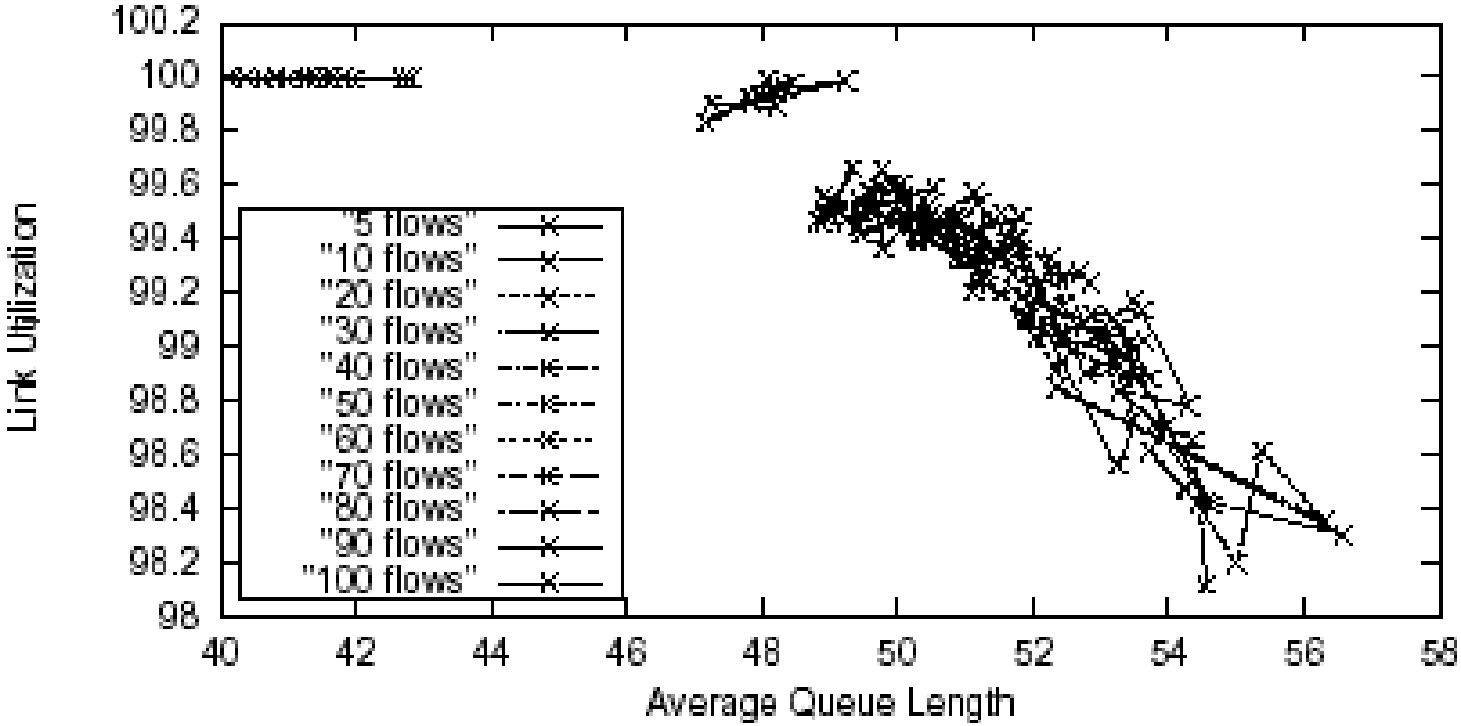


Figure 3: Delay-Utilization Tradeoff with Adaptive RED.

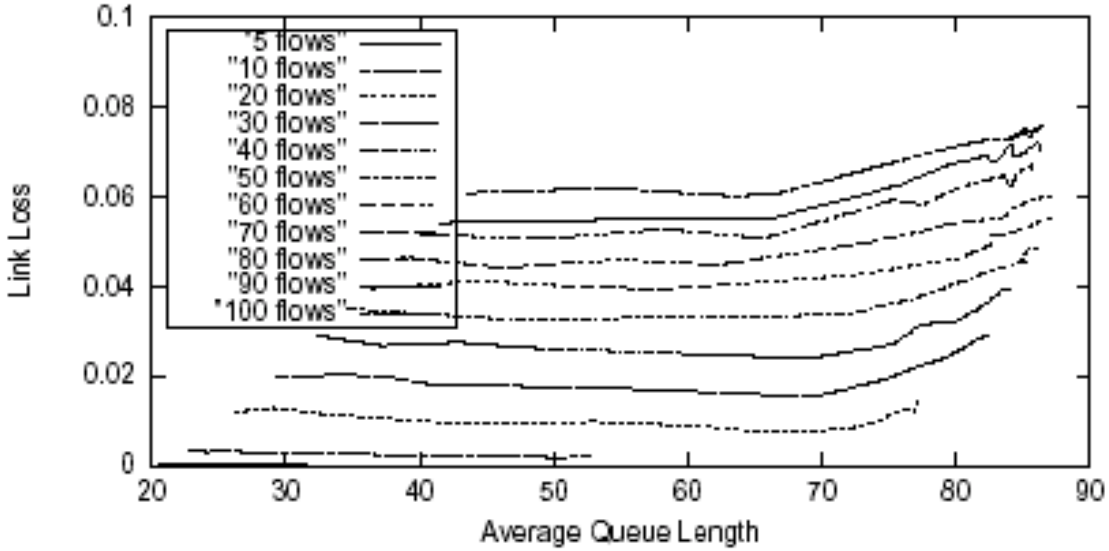


Figure 4: Delay-Loss Tradeoff with Normal RED, $w_q = 0.00026$.

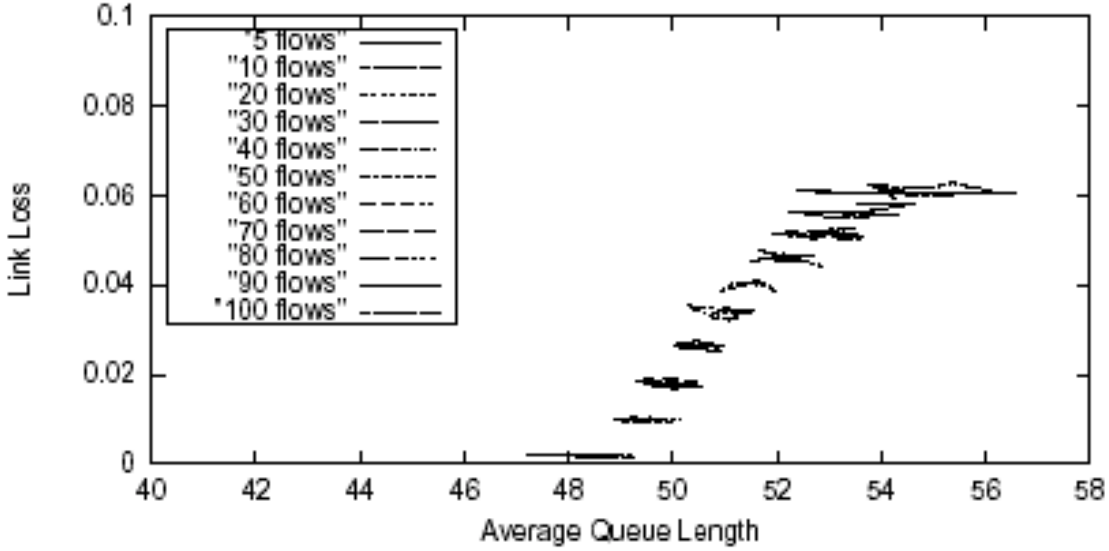


Figure 5: Delay-Loss Tradeoff with Adaptive RED.

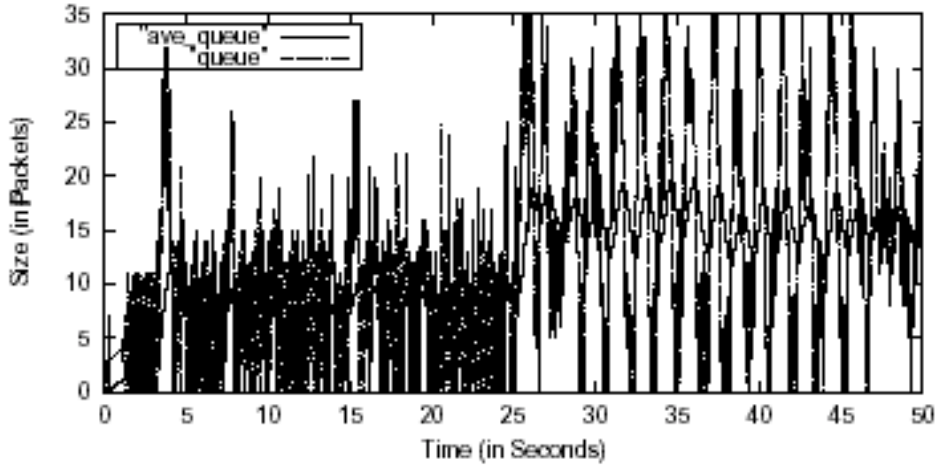
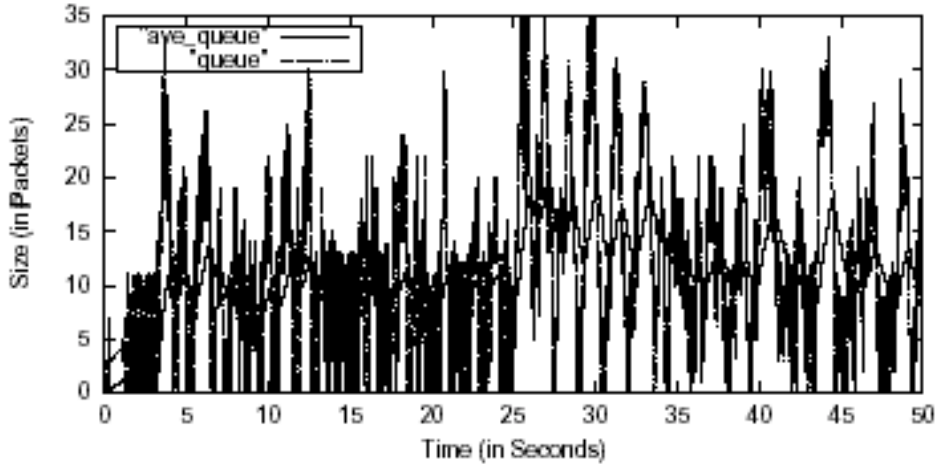


Figure 6: RED with an Increase in Congestion.



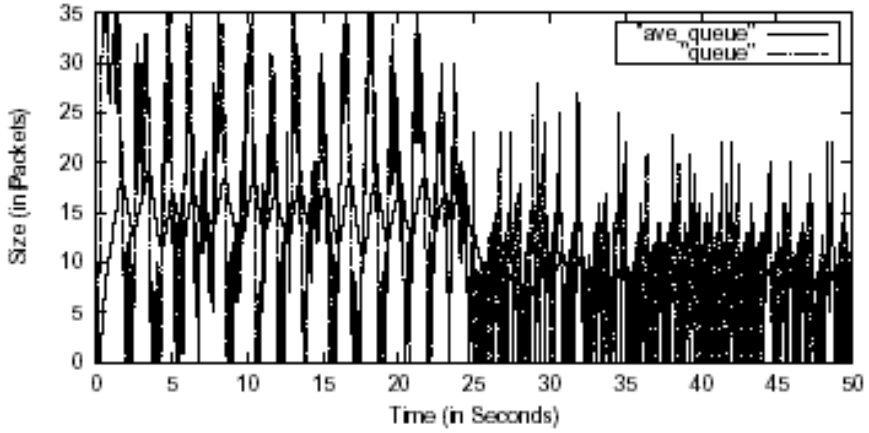


Figure 8: RED with a Decrease in Congestion.

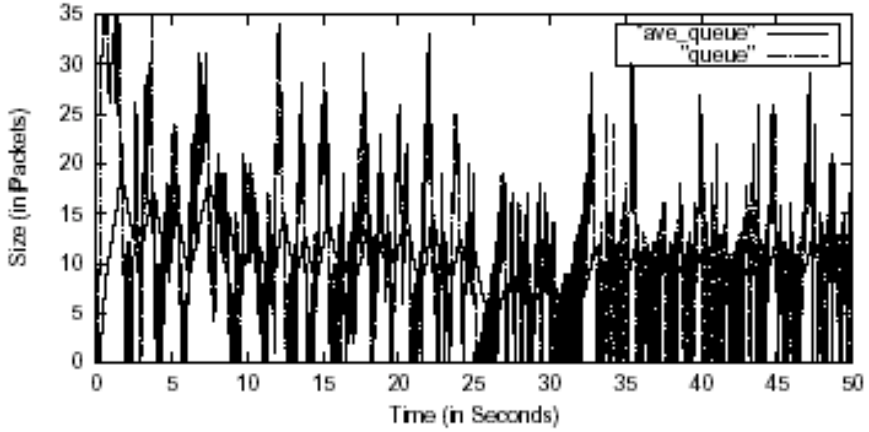
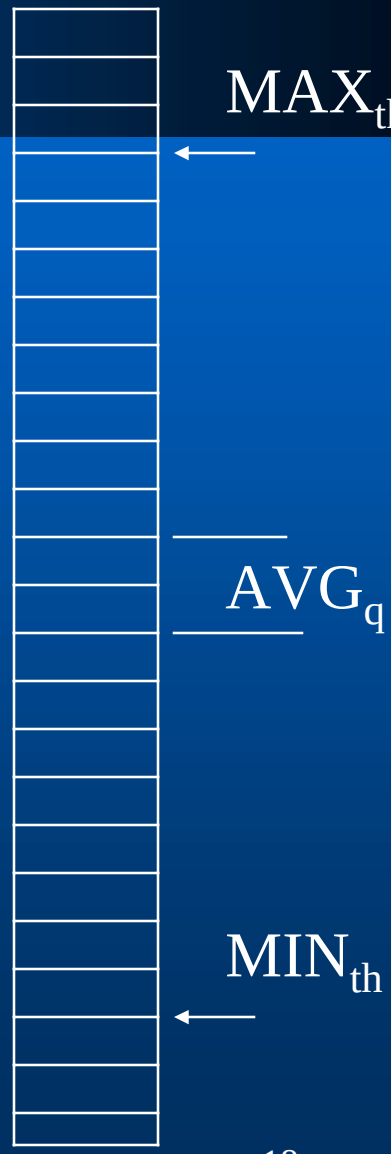


Figure 9: Adaptive RED with a Decrease in Congestion.

- Introduction
- Background and Related Work
- Metrics and Scenarios
- Pre-results
- Adaptive RED Algorithm.
- Parameters and their values
- Simulations.
  - Delay-Throughput Tradeoff.
- Conclusions.

Adaptive RED Algorithm

- MAX_p is adapted to keep AVG_q around $(MIN_{th} + MAX_{th})/2$
- AIMD is used for Adapting MAX_p
- Adaptation is slow – 0.5sec interval
- MAX_p bounded between [0.01,0.5]



Adaptive RED Algorithm

Every interval seconds(0.5sec):

If(AVG > Target and $MAX_p < 0.5$)

$MAX_p = MAX_p + \alpha$

Else if(AVG < Target and $MAX_p > 0.01$)

$MAX_p = MAX_p * \beta$

$\alpha = Min(0.01, MAX_p/4)$

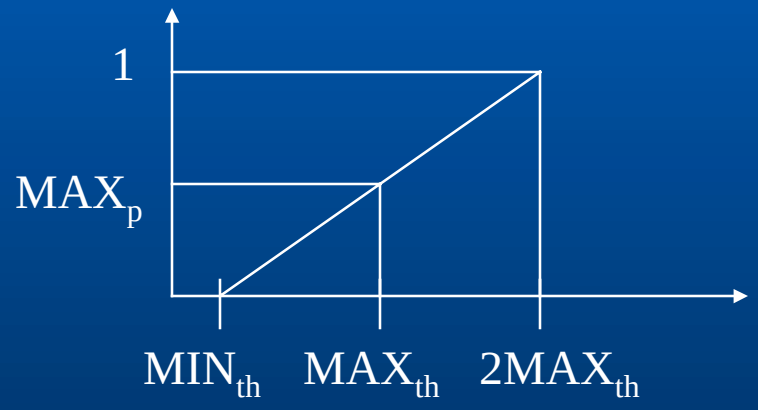
$\beta = 0.9$

- Introduction
- Background and Related Work
- Metrics and Scenarios
- Pre-results
- Adaptive RED Algorithm.
- Parameters and their values
- Simulations.
- Delay-Throughput Tradeoff.
- Conclusions.

MAX_p Range [0.01, 0.5]

Upper bound 0.5

- Not trying to optimize for packet drop rates more than 50%.
- Gentle RED



Lower bound 0.01

- No one will object lower delays
- Limits the MAX_p Range – Important as Adaptation is slow (0.5 sec interval)

Values of Increment α And decrement β

$$p = MAX_p \frac{(AVG - MIN_{th})}{(MAX_{th} - MIN_{th})}$$

$$AVG_1 = MIN_{th} + \frac{p}{MAX_p} (MAX_{th} - MIN_{th})$$

$$AVG_2 = MIN_{th} + \frac{p}{MAXp + \alpha} (MAX_{th} - MIN_{th})$$

Target Range > AVG2 – AVG1

$\alpha = \min(0.01, MAX_p/4)$

$\beta = 0.9$

MAX_{th} & W_q

- $MAX_{th} = 3 * MIN_{th}$ - As per latest recommendation of Sally Floyd. (They don't follow it in their simulations.)

- W_q gives a Time Constant in terms of packet arrival rate for AVG queue to adapt. ($-1/\ln(1-W_q)$) - Original RED

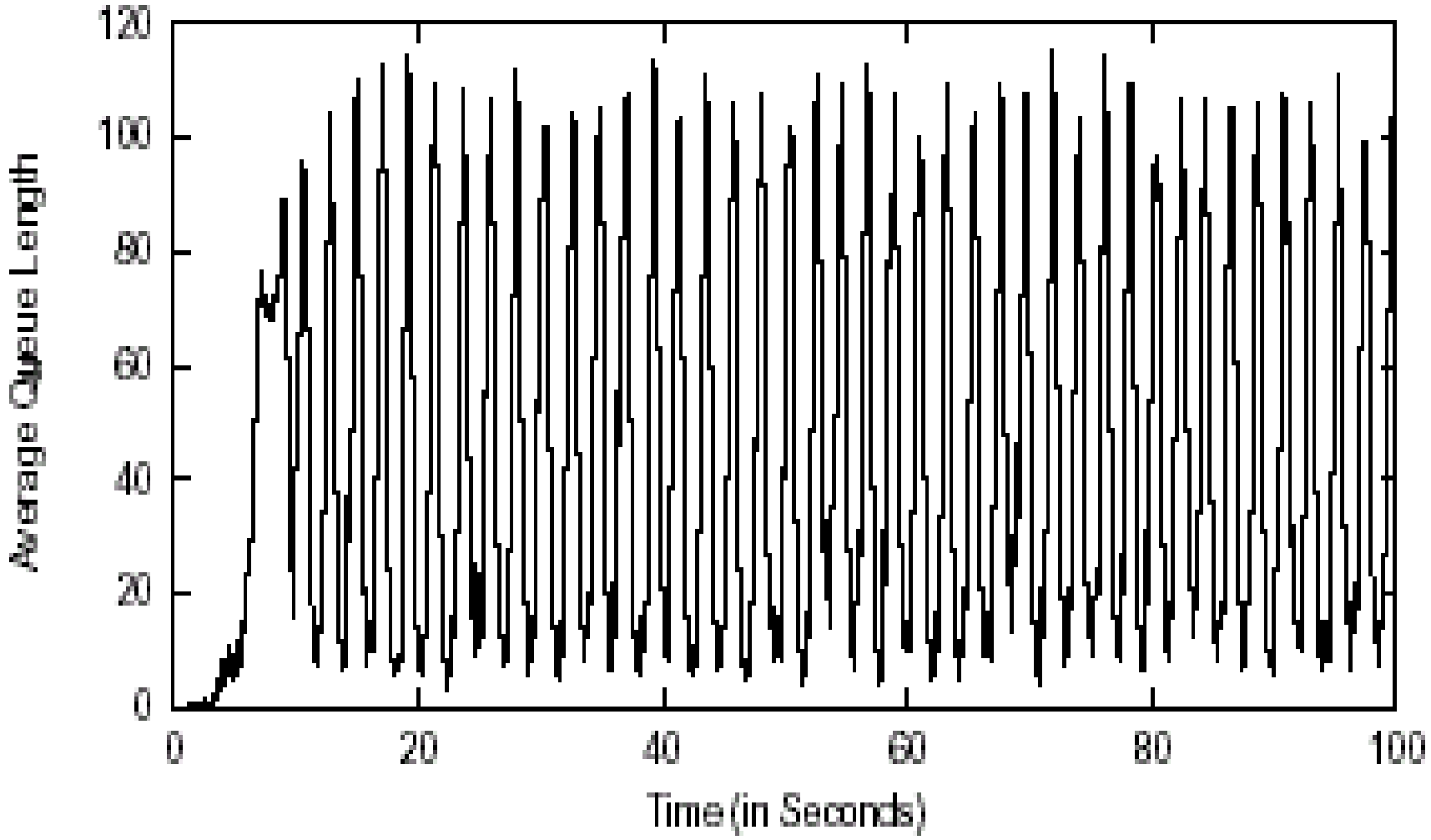
Since it is in terms of packet arrival rate, should be dependent on link capacity C.

$$C = -1/\ln(1-W_q) \quad \Rightarrow \quad W_q = 1 - \exp(-1/C)$$

- Introduction
- Background and Related Work
- Metrics and Scenarios
- Pre-results
- Adaptive RED Algorithm.
- Parameters and their values
- Simulations.
  - Delay-Throughput Tradeoff.
- Conclusions.

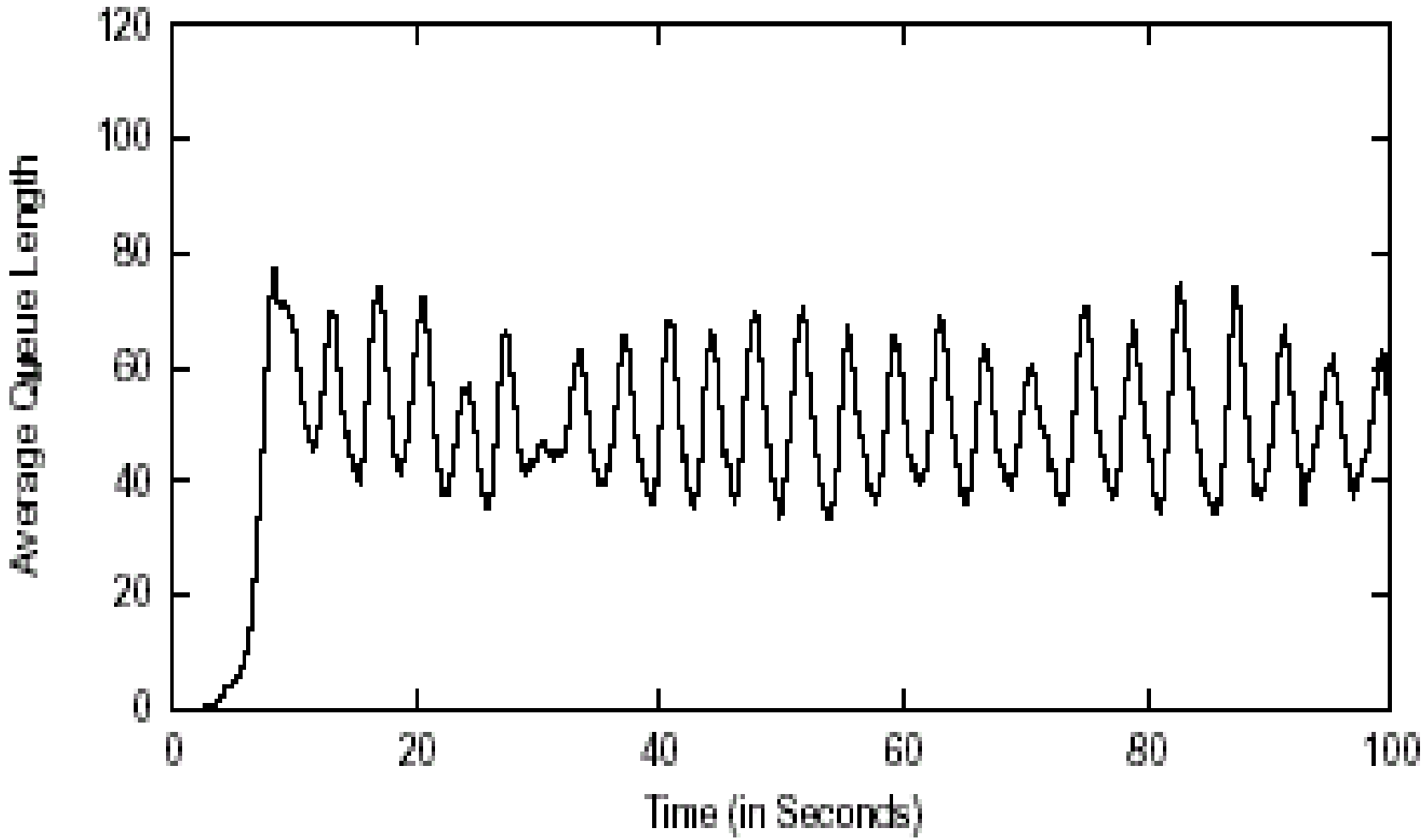
RED, one-way long-lived traffic, W_q= 0.002

100 long-lived flows, 250ms RTT, MIN_{th}=20, MAX_{th}=80



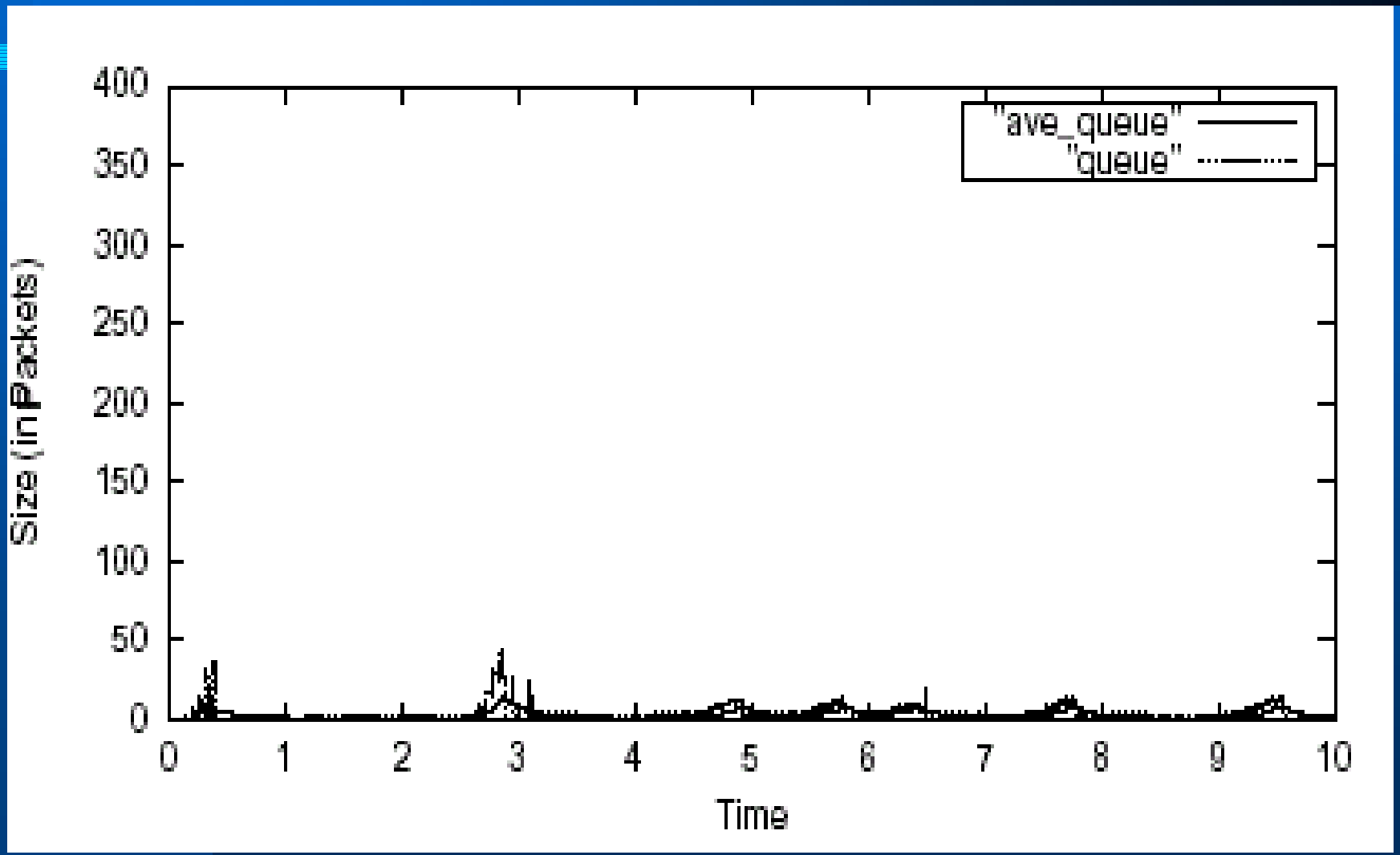
Adaptive-RED, one way long-lived traffic W_q = 0.00027

100 long-lived flows, 250ms RTT, MIN_{th}=20, MAX_{th}=80



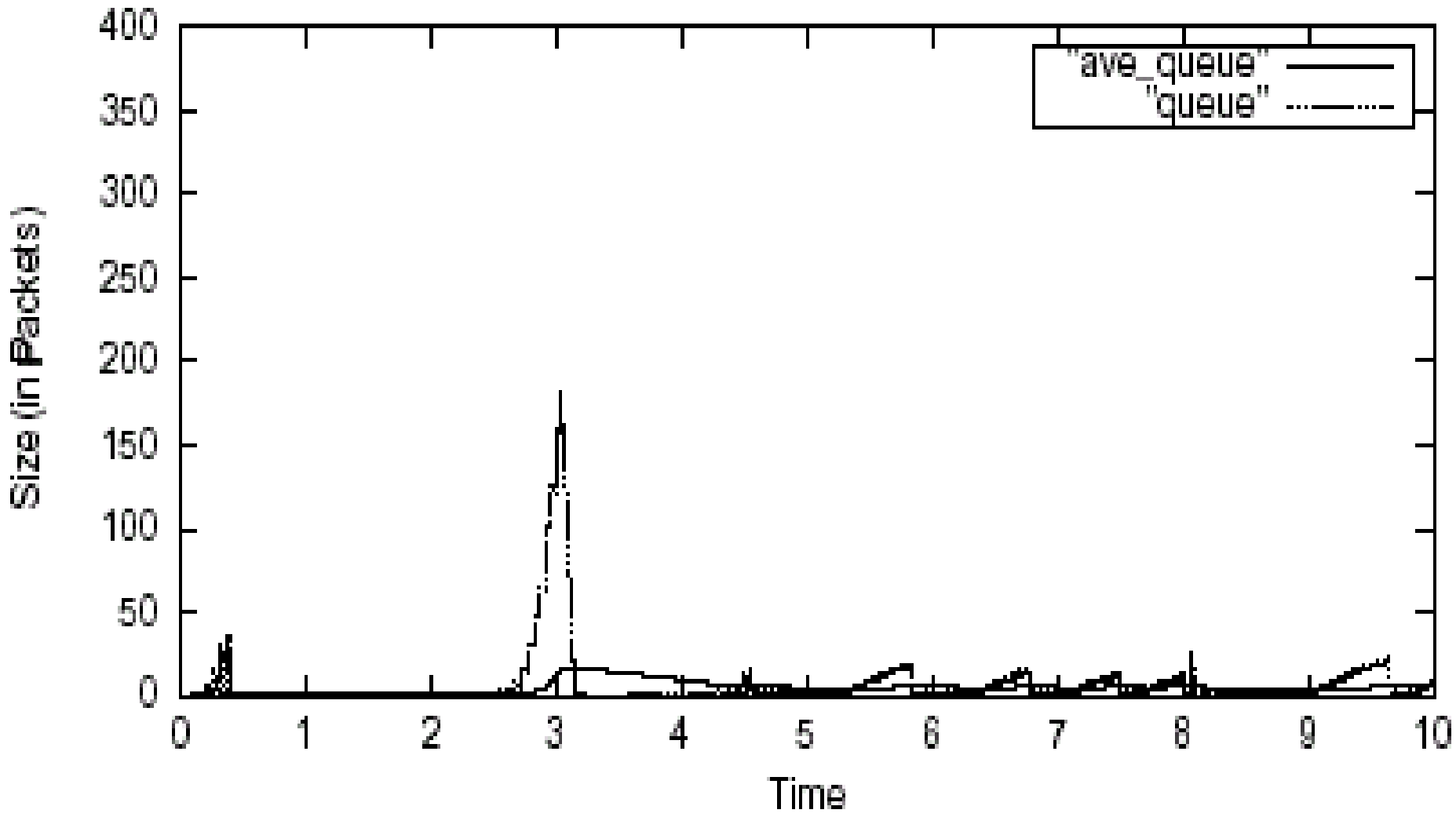
RED, two flows, W_q = 0.002 (Large W_q)

2 TCP flows, 1st start at time 0, 2nd at 2.5sec



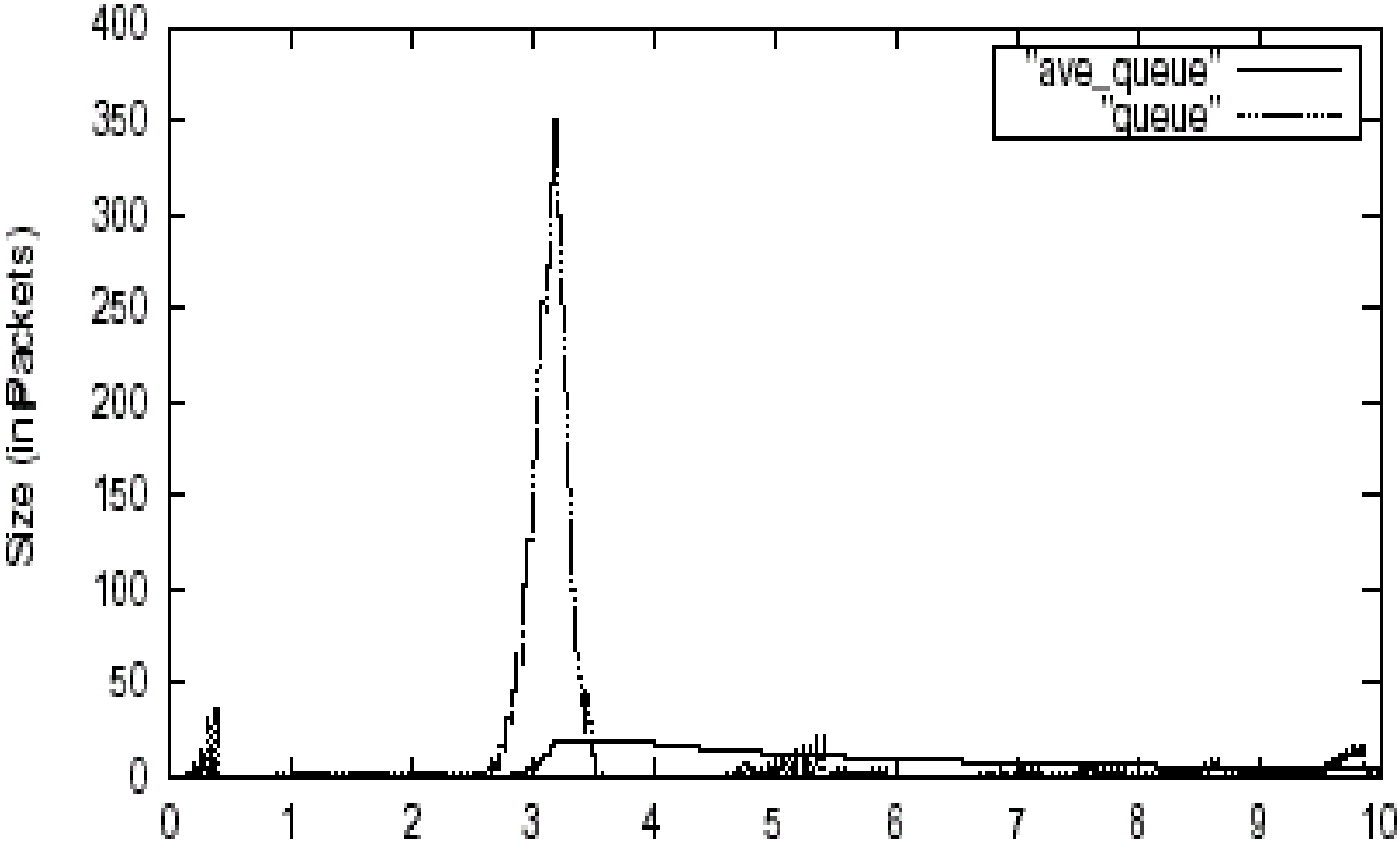
RED, automatic setting for W_q, 0.00027

2 TCP flows, 1st start at time 0, 2nd at 2.5sec



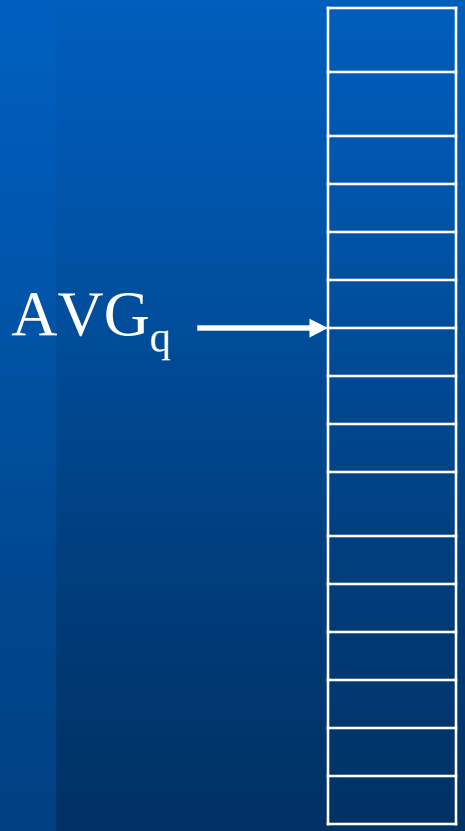
RED, W_q too small, 0.0001

2 TCP flows, 1st start at time 0, 2nd at 2.5sec



- Introduction
- Background and Related Work
- Metrics and Scenarios
- Pre-results
- Adaptive RED Algorithm.
- Parameters and their values
- Simulations.
- Delay-Throughput Tradeoff.
- Conclusions.

Delay-Throughput Tradeoff



$$AVG_q = \frac{MIN_{th} + MAX_{th}}{2}$$

$$MAX_{th} = 3 * MIN_{th}$$

$$AVG_q = 2* MIN_{th}$$

$$Delay_{target} * C = 2 * MIN_{th}$$

$$MINth = \frac{Delay_{target} * C}{2}$$

- Introduction
- Background and Related Work
- Metrics and Scenarios
- Pre-results
- Adaptive RED Algorithm.
- Parameters and their values
- Simulations.
- Delay-Throughput Tradeoff.
- Conclusions.

Conclusions

- Reduces RED's parameter sensitivity.
- Network operators can configure delay by using proper value for $MIN_{th.}$