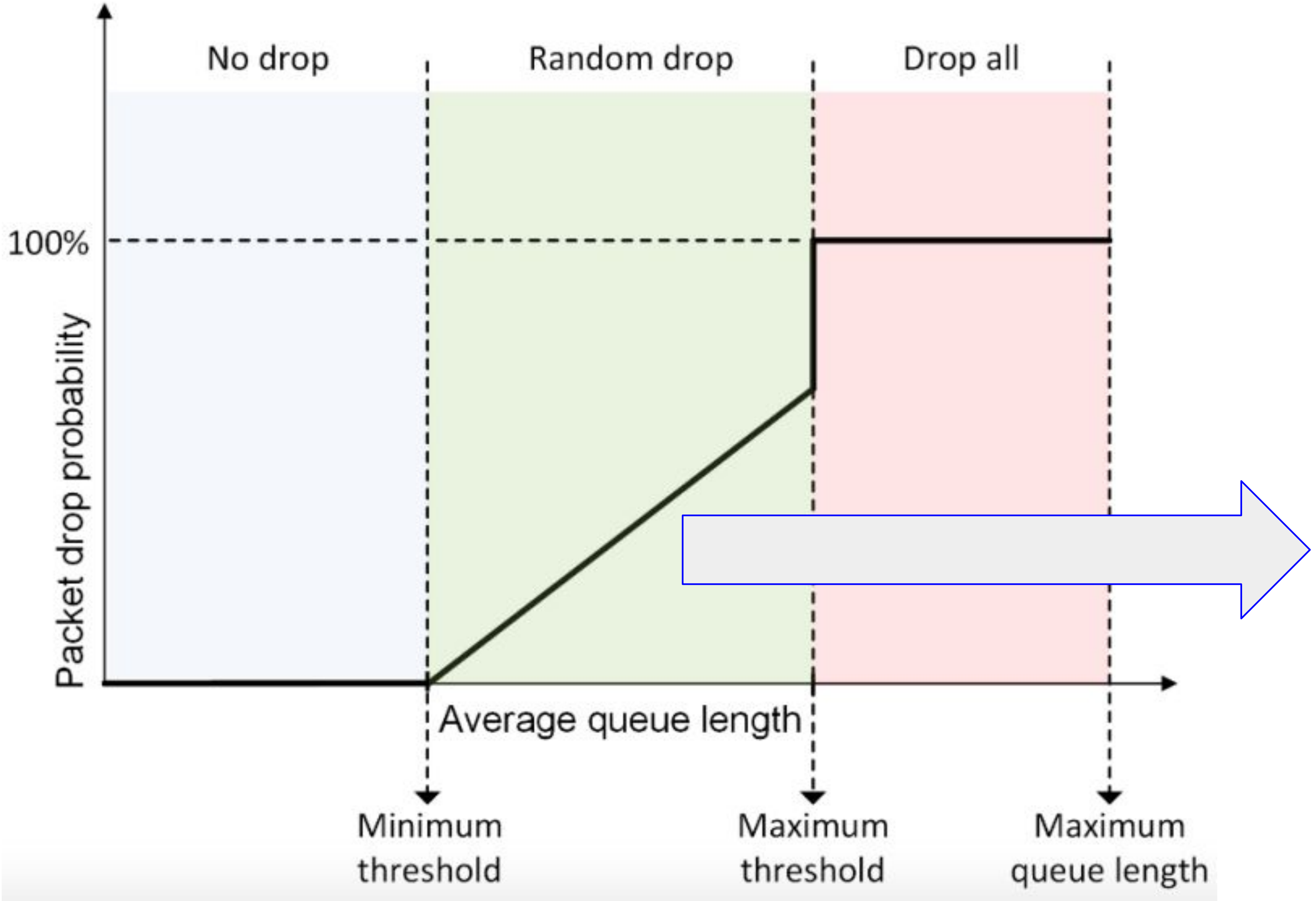
Adaptive RED Queue Discipline

Mohit P. Tahiliani
Assistant Professor
Department of Computer Science and Engineering
National Institute of Technology Karnataka, Surathkal, India
tahiliani@nitk.edu.in

Motivation: Adaptive RED



Instead of increasing the P_d linearly, it might be better if P_d is increased slowly when it is near to min_{th} and increased sharply when it is near to max_{th}.

One of the solutions: Adapt max_p

Image Credits: http://ce.sc.edu/cyberinfra/workshops/Material/NTP/Lab%2016.pdf

Main contributions in Adaptive RED paper

- Automatic setting of minimum threshold (min_{th})
  - It is set as a function of the link capacity (C) and target queue delay
- Automatic setting of maximum threshold (max_{th})
  - It is set depending on the value of min_{th}
- Automatic setting of w_q
  - It is set as a function of the link capacity (C)
- Adaptive setting of max_p
  - It is adapted according to the current average queue length

Adaptive RED vs Self Configuring RED

- max_p is adapted not just to keep the average queue size between min_{th} and max_{th}, but to keep the average queue size within a 'target range' halfway between min_{th} and max_{th}. Example: if min_{th} = 5 packets and max_{th} = 15 packets, then:

  target range = [min_{th} + 0.4 x (max_{th} - min_{th}), min_{th} + 0.6 x (max_{th} - min_{th})]

  Hence, target range = [5 + 0.4 (15 - 5), 5 + 0.6 (15 - 5)] = [9, 11]
- max_p is adapted slowly, over time scales greater than a typical round-trip time, and in small steps
- max_p is constrained to remain within the range [0.01, 0.5] (i.e., 1% to 50%)
- AIMD policy is used to adapt max_p, unlike MIMD policy which is used in Self Configuring RED

Automatic setting of minimum threshold (min_{th})

- What happens if the min_{th} is set to a low value?
  - Degradation of throughput
- What happens if the min_{th} is set to a high value?
  - Queue delay increases
- What is the best approach to estimate a suitable value for the min_{th}?
  - Set the min_{th} to be a function of the link capacity (C). Why?
    - If the link is slow, incorrect setting of min_{th} can lead to high queuing delays
    - If the link is fast, incorrect setting of min_{th} can lead to loss of throughput
  - Decide upon a suitable 'target queue delay' (i.e., acceptable queue delay)
    - Set the min_{th} to be a function of the target queue delay

Automatic setting of minimum threshold (min_{th})

- min_{th} is calculated as:

  $$min_{th} = (target\_queue\_delay \times C) \div 2$$ // Question: Why divide by 2?

  where,

  C = capacity of the link in packets (can be obtained by: Bandwidth ÷ packet size)

  target_queue_delay is 5ms (is a user configurable parameter)

- Sally Floyd's recommendation to set the min_{th} automatically is:

  $$min_{th} = \max\,[5, (target\_queue\_delay \times C) \div 2]$$

  - min_{th} of 5 packets was found to work well for low and moderate link capacity
    - So min_{th} of at least 5 packets is recommended to ensure that the throughput is not affected.

Automatic setting of maximum threshold (max_{th})

- max_{th} is calculated as:

$$max_{th} = 3 \times min_{th}$$

- This ensures that the 'target range' for average queue size is $2 \times min_{th}$
- Example: if $min_{th} = 5$ packets and $max_{th} = 15$ packets, then:

$$\text{target range} = [min_{th} + 0.4 \times (max_{th} - min_{th}),\ min_{th} + 0.6 \times (max_{th} - min_{th})]$$

Hence, target range = $[5 + 0.4\ (15 - 5),\ 5 + 0.6\ (15 - 5)] = [9, 11]$ // this is $2 \times min_{th}$

Automatic setting of w_q

- w_q is set to be a function of the link capacity (C):

  $w_q = 1 - e^{(-1/C)}$ // Verify whether this is same in ns-2/ns-3 implementation

  where, C is the link capacity in packets/second (i.e., Bandwidth ÷ packet size)
- If the queue size changes from one value (old) to another (new), it takes "$-1 / \ln(1 - w_q)$" packet arrivals for the 'average queue size' to reach 63% of the 'new queue size'
- Thus, "$-1 / \ln(1 - w_q)$" is referred to as a 'time constant' of the estimator for the average queue size (but it is specified in packet arrivals, and not actually in time).
- Example: if $w_q = 0.002$, it corresponds to 500 packet arrivals.
  - But suppose if the bandwidth available is 1Gbps, 500 packet arrivals account for a small amount of time. Hence, even smaller values of w_q would be better.

The Adaptive RED algorithm (adapting max_p)

Every *interval* seconds:
    if $(avg > target$ and $max_p \leq 0.5)$
        increase max_p:
        $max_p \leftarrow max_p + \alpha$;
    elseif $(avg < target$ and $max_p \geq 0.01)$
        decrease max_p:
        $max_p \leftarrow max_p * \beta$;

Variables:
avg: average queue size

Fixed parameters:
interval: time; 0.5 seconds
target: target for *avg*;
    $[min_{th} + 0.4 * (max_{th} - min_{th})$,
        $min_{th} + 0.6 * (max_{th} - min_{th})]$.
α: increment; min(0.01, $max_p/4$)
β: decrease factor; 0.9

Important Note!

- In Adaptive RED, **α** is used to increment the value of max_p, whereas in Self Configuring RED, it is used to decrement the value of max_p.
- In Adaptive RED, **β** is used to decrement the value of max_p, whereas in Self Configuring RED, it is used to increment the value of max_p.

Image Credits: https://www.icir.org/floyd/papers/adaptiveRed.pdf

Deriving bounds for **α** in Adaptive RED

$$p = max_p \times (\frac{avg - min_{th}}{max_{th} - min_{th}})$$ Eq. (1)

Before adapting max_p

$$avg_1 = min_{th} + \frac{p}{max_p} \times (max_{th} - min_{th})$$ Eq. (2)

and after adapting max_p

$$avg_2 = min_{th} + \frac{p}{max_p + \alpha} \times (max_{th} - min_{th})$$ Eq. (3)

Subtracting

$$avg_1 - avg_2 = \frac{\alpha}{max_p + \alpha} \times \frac{p}{max_p} \times (max_{th} - min_{th})$$ Eq. (4)

Hence to ensure *avg* does not exceed *above target* to *below target*

Deriving bounds for α in Adaptive RED

$$\frac{\alpha}{max_p + \alpha} < 0.2 \qquad \text{Eq. (5)}$$

$$\alpha < 0.25\ max_p \qquad \text{Eq. (6)}$$

Deriving bounds for **β** in Adaptive RED

Similarly for β, before adapting max_p

$$avg_1 = min_{th} + \frac{p}{max_p} \times (max_{th} - min_{th})$$ Eq. (7)

and after adapting max_p

$$avg_2 = min_{th} + \frac{p}{max_p \times \beta} \times (max_{th} - min_{th})$$ Eq. (8)

Subtracting

$$avg_1 - avg_2 = \frac{1-\beta}{\beta} \times \frac{p}{max_p} \times (max_{th} - min_{th})$$ Eq. (9)

Hence to ensure *avg* does not exceed *below target* to *above target*

Deriving bounds for **β** in Adaptive RED

$$\frac{1-\beta}{\beta} < 0.2 \qquad \text{Eq. (10)}$$

$$\beta > 0.83 \qquad \text{Eq. (11)}$$

Question 1:

Suppose the target range defined in Adaptive RED is modified as:

$$\text{target range} = [min_{th} + 0.48 \times (max_{th} - min_{th}),\ min_{th} + 0.52 \times (max_{th} - min_{th})]$$

What will be the new bounds for **α** and **β**?

Question 2:

How many knobs are removed in Adaptive RED and how many new knobs are added?

Recommended Reading

Adaptive RED:

Link: https://www.icir.org/floyd/papers/adaptiveRed.pdf