Proportional Integral (PI) Controller and PI Controller Enhanced (PIE) Queue Disciplines

Mohit P. Tahiliani

Assistant Professor

Department of Computer Science and Engineering

National Institute of Technology Karnataka, Surathkal, India

tahiliani@nitk.edu.in

Overview

- Proportional Integral (PI) controller was designed to overcome the problems of RED
  - It uses ‘instantaneous queue length’ as a congestion metric
- PI Controller Enhanced (PIE) queue discipline [RFC 8033]
  - A popular variant of PI Controller.
  - PIE uses queue delay as a congestion metric, like CoDel
  - Implemented in the Linux kernel
- Flow Queue PIE (FQ-PIE) queue discipline
  - A popular variant of PIE implemented in the Linux kernel
- DOCSIS PIE queue discipline [RFC 8034]
  - A variant of PIE developed for DOCSIS standard.

Working of PI and PIE

- PI and PIE operate during the ‘enqueue’ time
  - Note: do not confuse this with ‘input port’ in the router architecture
    - PI and PIE run on the ‘output port’, but during the ‘enqueue’ time!
- PI and PIE ‘do not’ operate on arrival of every packet like RED does
  - PI runs once in every 6ms (w) and PIE runs once in every 15 ms (t_update)
- PI and PIE decide whether the incoming packet should be enqueued or dropped
  - PI and PIE algorithm contain the following components:
    - Calculation of current queue length (PI) / instantaneous queue delay (PIE)
    - Calculation of drop probability
    - Decision making logic (helps to decide whether the incoming packet should be enqueued or dropped)

Working of PI and PIE

1. Calculation of current queue length (PI)
   - Every ‘w’ ms, PI algorithm fetches the information regarding the current queue length (cur_qlength). It’s a simple function/method in implementations.
1. Calculation of current queue delay (PIE)
   - Every ‘t_update’ ms, PIE algorithm calculates the queue delay.
   - Two ways to estimate current queue delay (cur_qdelay):
     i. Little’s Law as shown in Eq. (1) [recommended default in RFC 8033]

$$\text{cur\_qdelay} = \text{cur\_qlength} / \text{avg\_departure\_rate} \qquad \text{Eq. (1)}$$

     ii. Using timestamps like CoDel, as shown in Eq. (2) [implemented in Linux]

$$\text{cur\_qdelay} = \text{dequeue\_time} - \text{enqueue\_time} \qquad \text{Eq. (2)}$$

Working of PI and PIE (contd ...)

2. Calculation of drop probability (PI)

   ○ Drop probability is calculated as shown in the equation below:

drop_prob = a (cur_qlength – target) – b (old_qlength – target) + drop_prob

Eq. (3)

2. Calculation of drop probability (PIE)

   ○ Drop probability is calculated as shown in the equation below:

drop_prob = a (cur_qdelay – target) + b (cur_qdelay – old_qdelay) Eq. (4)

Working of PI and PIE (contd ...)

3. Decision making logic

Once the ‘drop_prob’ is calculated, PI and PIE use the following logic to decide whether the incoming packet must be enqueued or dropped:

```
if (drop_prob ≤ R)
    enqueue the incoming packet
else
    drop the incoming packet
```

where, R = uniformly distributed random number generated between [0, 1]

Note: It is important that a well known random number generator is used to generate R. If the implementation of the random number generator is not correct, PI and PIE’s performance might get affected.

Additional features in PIE

1. Burst allowance (allows small bursts to pass by without getting punished)

2. Auto-tuning the drop_prob

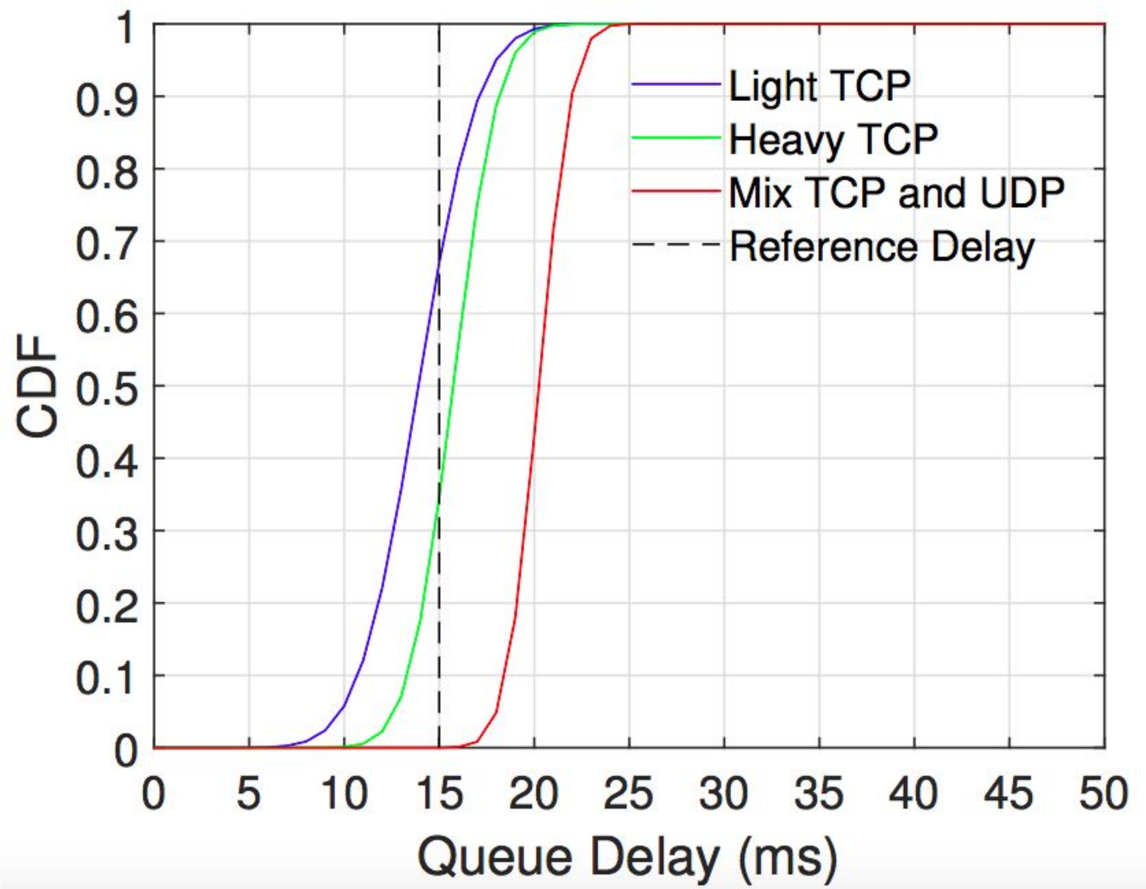
3. Avoids a sharp rise in drop_prob

4. Decays drop_prob when queue is idle

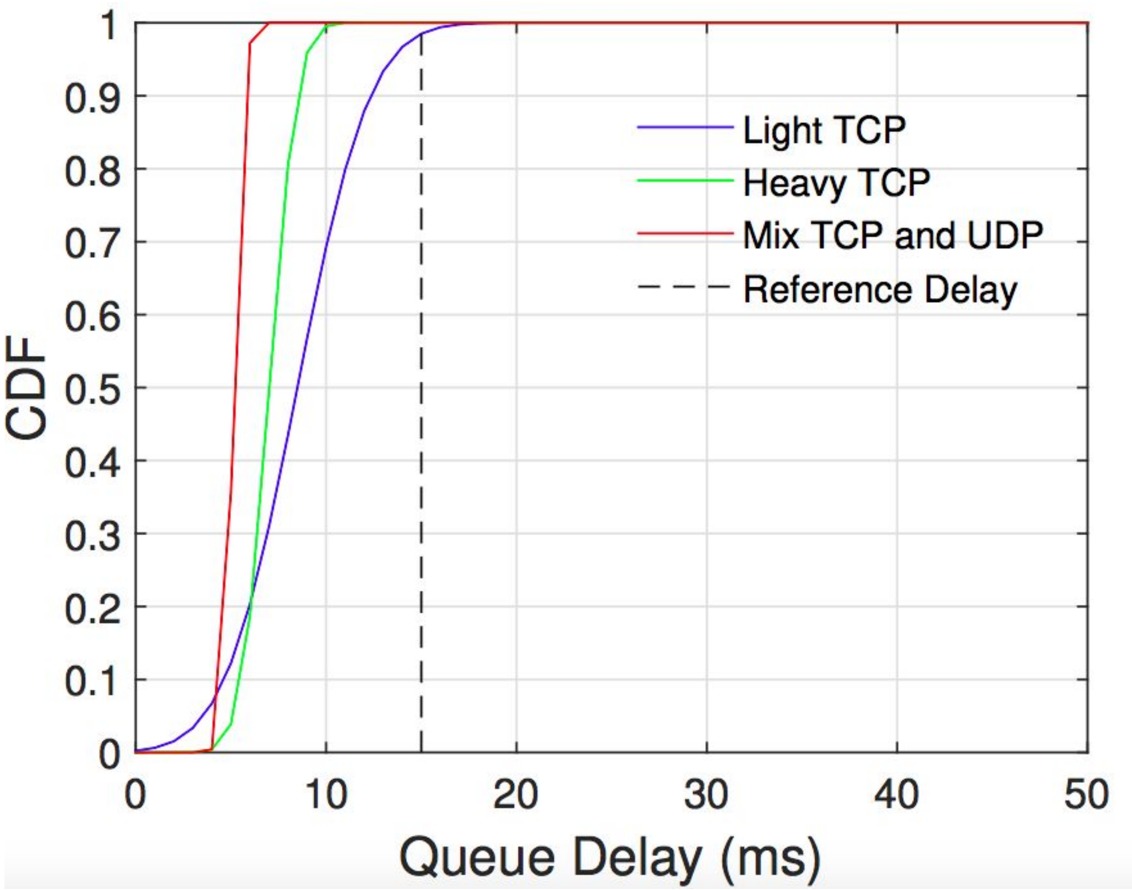
5. Activating / deactivating PIE depending on current queue length

Minstrel PIE

- Adapts “target” depending on the network load



PIE



Minstrel PIE

Recommended Reading

RFC 8033: Proportional Integral Controller Enhanced (PIE): A Lightweight Control Scheme to Address the Bufferbloat Problem (Link: https://datatracker.ietf.org/doc/html/rfc8033)

Patil, S.D. and Tahiliani, M.P., 2019. Minstrel PIE: Curtailing Queue Delay in Unresponsive Traffic Environments. Computer Communications, 139, pp. 16–31.

Imputato, P., Avallone, S., Tahiliani, M.P. and Ramakrishnan, G., 2020. Revisiting design choices in queue disciplines: The PIE case. Computer Networks, 171, pp. 107–136.

The problem identified in PIE implementation of Linux: https://youtu.be/nJ07FGmZ3ig?t=3980

Ramakrishnan, G., Bhasi, M., Saicharan, V., Monis, L., Patil, S.D. and Tahiliani, M.P., 2019, October. FQ-PIE queue discipline in the Linux Kernel: Design, implementation and challenges. In 2019 IEEE 44th LCN Symposium on Emerging Topics in Networking (LCN Symposium) (pp. 117–124). IEEE.