Packet Scheduling Algorithms for QoS in Internet

By

Abhay Karandikar
Department of Electrical Engineering
Indian Institute of Technology, Bombay

karandi@ee.iitb.ac.in

The World of TCP

- TCP congestion control is based on Additive Increase and Multiplicative Decrease (AIMD) algorithm

- AIMD mecthod is biased towards connection with smaller round trip-times

- In a router with FIFO scheduling, the ones with smaller round trip times are likely to grab most of the link capacity

- Need for a fair scheduling mechanism at a packet switch or router

Functions of Scheduling

(S Keshav: An Engineering Approach...)

- Two Orthogonal Components
  - Decides the order of servicing packets
  - Manages the queue of packets
- To satisfy diverse objectives
  - Share the available bandwidth of a link fairly
  - Provide network level QoS
  - Hierarchical link sharing
- Work Conserving and Non Work Conserving Scheduler

The Scheduling Conservation Law

- FCFS can not differentiate among connections

- Non FCFS can provide differential services

- Kleinrock Conservation Law holds good for Non FCFS. Let $\rho_i = \lambda_i x_i$ be the mean utilization due to connection i. Here λ_i and x_i are the arrival rate and service time of connection i at server servicing N connections and q_i be the queueing time.

$$\sum_{i=1}^{N} \rho_i q_i = \text{Constant}$$

Attributes of Scheduling

- Implementation Simplicity
- Fairness and Flow Isolation
- Scalability
- Performance Bounds
- Simple Admission Control

Fluid Flow Fair Queueing

- Assume the traffic to be fluid
- Visit each non empty queue
- Serve infinitesimal amount from each queue
- This service order results in max-min fairness
- Connections can be given weights and receive service in proportion to their weights
- Hence it is called Generalized Processor Sharing (GPS)

GPS- Formal Definition

(Parekh and Gallager)

- Assume the traffic to be fluid
- Let ρ_i be the rate allocated to session i
- Let $W_i(\tau, t)$ =amount of service received by session i during (τ, t)
- Fluid fair queueing or GPS is defined as the service discipline where for sessions i and j which are continously backlogged during (τ, t),

$$W_i(\tau, t)/\rho_i = W_j(\tau, t)/\rho_j$$

Weighted Fair Queueing (WFQ)

(Demers, Keshav and Shenker)

- Also known as Packetized GPS (PGPS)
- The GPS server is modeled as doing bit by bit round robin service.
- WFQ simulates GPS on the side
- Determine when the last bit of packet would have been served. Call this Finish Number (Virtual Finish Time)
- Serve packets in the increasing order of their finish numbers

Concept of Virtual Time

- Consider a busy period beginning at $t = 0$. Consider $(\tau, t]$ to be any subinterval of the busy period

- The virtual time $V(t)$ is defined as the function which satisfies the following
$$V(t) - V(\tau) = \frac{W_i(\tau, t)}{\rho_i}$$
for any connection i that was backlogged in (τ, t)

- The following result can be proved
$$V(t) - V(\tau) = \frac{r(t - \tau)}{\sum_{i \in B(\tau,t)} \rho_i}$$

- Virtual Time is a piecewise linear function with a slope inversely proportional to the number of backlogged connections

WFQ- Formal Definition

- Let a packet p_i^k arrives at time a_i^k at session i
- Let d_i^k denote the time when the packet finishes service in GPS
- Define $F_i^k = V(d_i^k)$
- The following relation can be proved

$$F_i^k = L_i^k/\rho_i + \max(F_i^{k-1}, V(a_i^k))$$

- F_i^k are Virtual Finish Times.
- Packets are served in the increasing order of F_i^k

Computing Virtual Time

- Computing $V(t)$ requires keeping track of the backlogged sessions

- Complexity in evaluating $V(t)$ depends on the frequency of breakpoints in $V(t)$

- Number of breakpoints in $V(t)$ in an abritrarily short period of time can approach the total number of sessions

Is WFQ fair ?

- d^k= time at which p^k departs in GPS
- $\hat{d}^k$ = time at which p^k departs in WFQ
- We can prove the following relation

$$\hat{d}^k - d^k \leq L_{max}/r$$

Self Clocked Fair Queueing (SCFQ)

- Approximates the Virtual Time Computation

- Note that $F_i^k = V(d_i^k)$

- This suggests that we can estimate the virtual time by the finish tag of the packet currently under service

- Let a packet p_i^k be stamped with $\tilde{F}_i^k$

$$\tilde{F}_i^k = L_i^k/\rho_i + \max(\tilde{F}_i^{k-1}, \tilde{V}(a_i^k)$$

- Define $V(t) = \tilde{F}_i^\ell$ where $\tilde{F}_j^\ell$ is the time stamp of the packet currently under service and $\tilde{s}_j^\ell < t \leq \tilde{d}_j^\ell$

How fair is this algorithm ?

- Relative Fairness (defined by Golestani)
  - Difference in normalized service received by two sessions in a server

- In GPS, normalized service received by two sessions that are backlogged is same at every instant of backlogged period

- Not fair to just compare the normalized service

- Service missed by the session while being absent should be accounted for

Session's Virtual Time

- It is defined as

$$V_i^S(t) = u_i^S(0,t) + \frac{W_i(0,t)}{\rho_i}$$

- The service lag is defined as

$$\delta_i^S = V^S(t) - V_i^S(t)$$

- In GPS, the service lag is zero

- In SCFQ, we have $\delta_i(t) \leq L_{max}/\rho_i$

- Using this property, it can be proved that

$$|W_i(\tau,t)/\rho_i - W_j(\tau,t)/\rho_j| \leq L_i^{max}/\rho_i + L_j^{max}/\rho_j$$

System and Session Virtual Time

- System virtual time keeps track of the total work done in the system

- Session virtual time increases by the amount of the normalized service during session's backlogged period

- In GPS, session's virtual time is always equal to the system virtual time

- In SCFQ, the difference between the session's virtual time and system's virtual time is always bounded

Rate Proportional Servers

(Stiliadis and Verma)

- GPS offers the same normalized service to all backlogged sessions at every instant of time

- Introduce System Potential and Session Potential

- System Potential- Estimate of the total work done in the system

- Backlogged Period– Session Potential increases by the normalized service it receives

- The objective of Fair algorithm is to equalize the potential of each backlogged session

System Potential

- The algorithms differ in the way in which session potentials are updated

- System Potential keeps track of the total work done in the system

- System Potential satisfies the following conditions

$$P(t_2) - P(t_1) \geq (t_2 - t_1)$$

$$P(t) \leq \min_{i \in B(t)} (P_i(t))$$

Session Potential

- $P_i(t)$ is a constant if the session is not backlogged

- When the session becomes backlogged at time τ then

$$P_i(\tau) = \text{max}(P_i(\tau^-), P(\tau^-))$$

- For every time $t > \tau$ such that session i remains backlogged during the whole interval we have

$$P_i(t) = P_i(\tau) + W_i(\tau, t)/\rho_i$$

- Among the backlogged sessions with minimal potentials are served

- Potentials of all sessions receiving service increase at the same rate

Maintaining System Potential

- System Potential evolves as a piecewise linear function of time with slope 1

- At specific intervals, system potential is recalibrated to remove any discrepancy between its values and the actual global state of the system

- The interval between successive calibration intervals are bounded to achieve bounded unfairness

- Several algorithms have been defined for different recalibration mechanisms.

QoS- Delay Bounds

- Maximum delay that a packet from a given session can experience

- Can be derived if
  - If the traffic is characterized by Leaky Bucket
  - If the scheduler belongs to Latency Rate server class

Leaky Bucket Traffic Characterization

- Two Parameters: Token Rate ρ (bytes or packets/sec) and Bucket Depth σ (bytes or packets)

- Tokens are added at rate ρ to the bucket

- If the bucket is full, then tokens are leaked out

- When a packet arrives, the bucket is decremented by ℓ tokens where ℓ is the length of the packet

- If there are not enough tokens, the packet is stored in the buffer

Leaky Bucket-Properties

- The output can have a long term average rate no more than ρ

- Let $A(t)$ be the amount of bytes generated by a source in an interval of t seconds for any possible value of t, then output of leaky bucket satisfies $W(t) \leq \sigma + \rho t$

- The Leaky Bucket controller delays the traffic the least among all FIFO controllers that outputs (σ, ρ) traffic

Session Busy Period

- Maximal interval $(\tau_1, \tau_2]$ such that

$$A_i(\tau_1, t) \geq \rho_i(t - \tau_1)$$

- Session busy period is same for every scheduling algorithm, while backlogged periods may not coincide

- Let τ be the beginning of jth busy period of session i

- Let $\hat{\tau}$ be the time at which the last bit arrived during jth busy period leaves the server

Latency Rate Server

- Latency Rate server is defined as the server for which the following relationship holds true for any $t \in (\tau, \hat{\tau}]$

$$W_{ij}(\tau, t) \geq \max(0, \rho_i(t - \tau - C_i))$$

- The minimum C_i satisfying this relationship is called Latency of the server

- In LR servers, the average rate of service offered by the scheduler to a busy session over every interval starting at Θ_i from the beginning of the busy period is at least equal to its reserved rate

- Intuitively, the latency signifies the worst case transmission delay seen by the first packet coming from session i after an idle period

Latencies of Schedulers

- GPS, PGPS and SCFQ are latency rate servers

- The latency of GPS is zero

- The latency of WFQ is given by

$$\Theta_i = L_i/\rho_i + L_{max}/r$$

- The latency of SCFQ is given by

$$\Theta_i = L_i/\rho_i + (N-1)L_{max}/r$$

Latency Rate Servers- Delay Bounds

- Let the input to a latency rate server be (σ, ρ)
- The worst case delay D_i is given by

$$D_i \leq \sigma_i/\rho_i + \Theta_i$$

- Delay through a network of K LR servers in series is given by

$$D_i \leq \sigma_i/\rho_i + \sum_{j=1}^{K} \Theta_i$$

- The delay in PGPS or WFQ is thus given by

$$D_i \leq \sigma_i/\rho_i + L_i/\rho_i + L_{max}/r$$

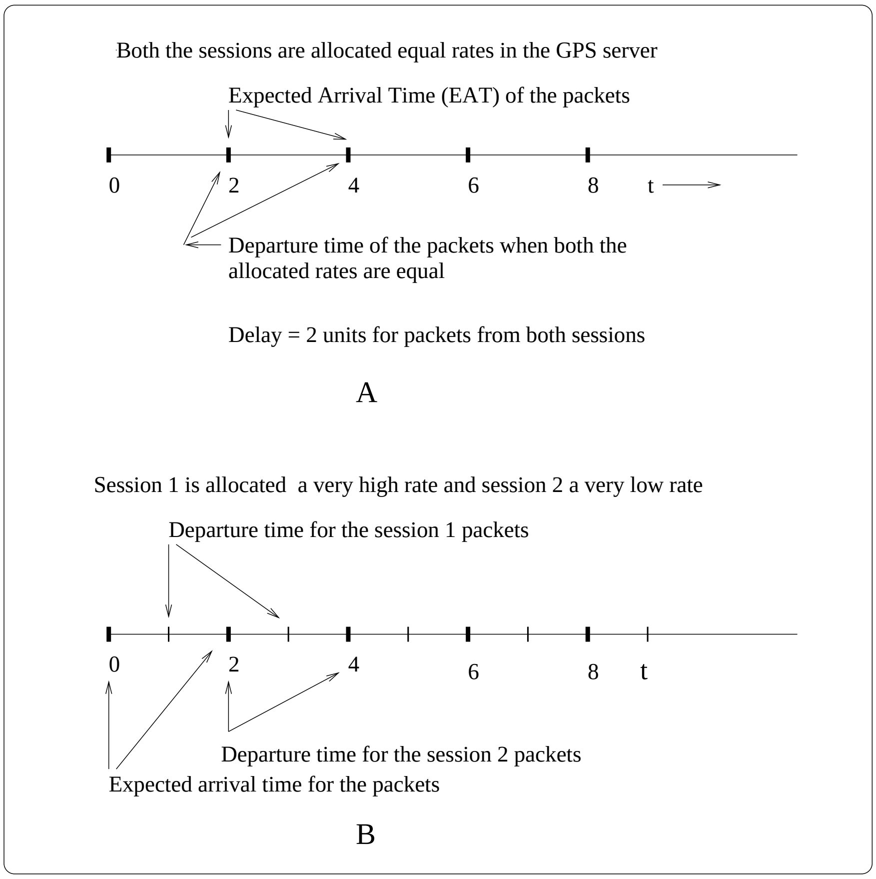
Delay Bandwidth Coupling

- $r = 2\,Mbps, L = 50\,bytes, \sigma_i = 2\ packets = 100\,bytes$.
  - For $\rho_1 = 32\,Kbps$,
  $$D_1 = 25\,ms$$
  - For $\rho_2 = 256\,Kbps$,
  $$D_2 = 3.12\,ms$$
- $L = 100\,bytes$
  - $D_1 = 50\,ms$
  - $D_2 = 6.25\,ms$

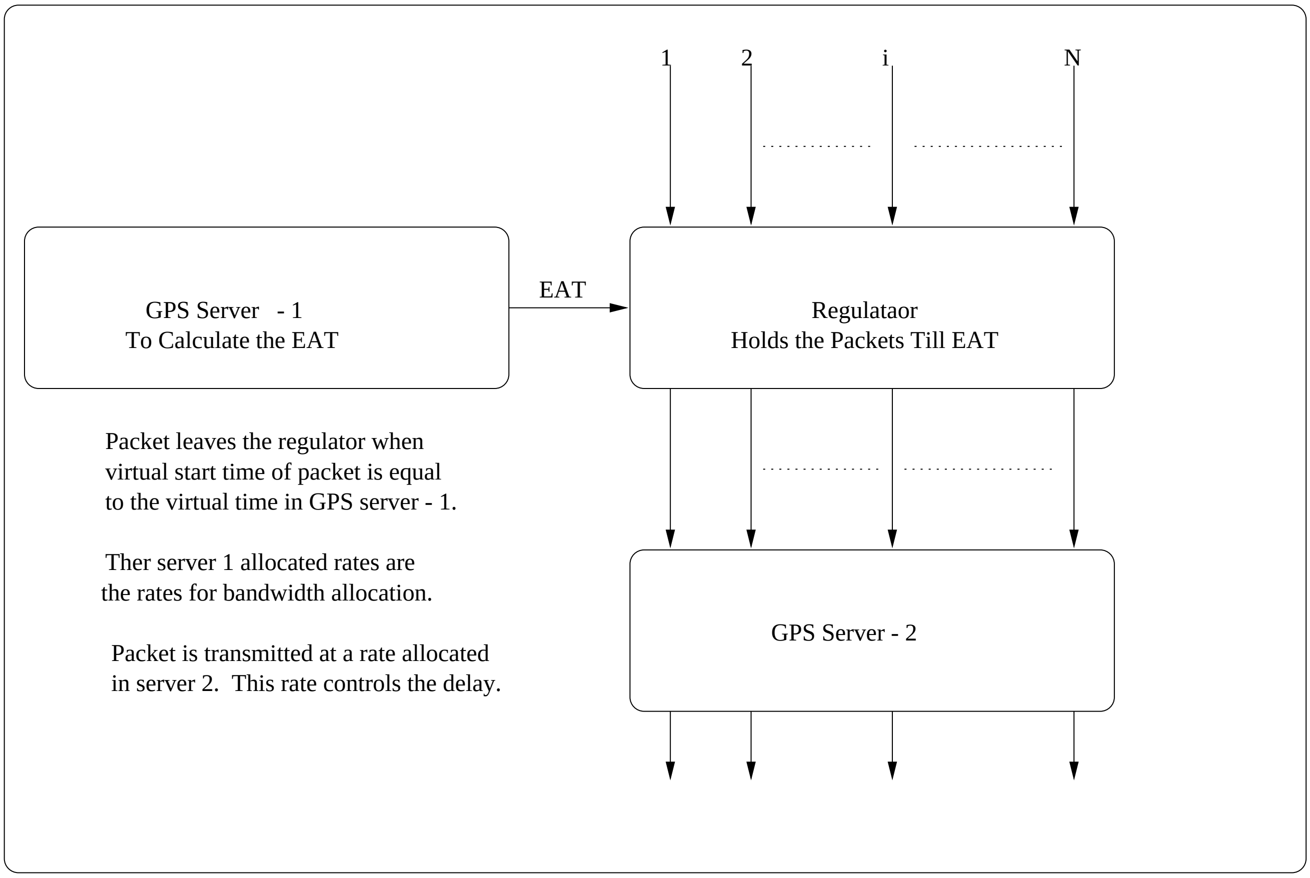
Delay Bandwidth Coupling

- Two sessions with $\rho_1 = \rho_2 = 1/2$, $r = 1$. Assume packet lengths=1

- Assume packets arrive at $t = 0, 2, 4, 6, \ldots$



Modified GPS



- GPS Server 1 - weights $\phi_i = \rho_i$

- GPS Server 2 - weights $\theta' s$

Packetized Version of Modified GPS

(Parikh and Karandikar)

- Packet p_i^k arrives at time a_i^k
- $S_i^k = MAX(F_i^{k-1}, v(a_i^k))$
- $F_i^k = S_i^k + \frac{L_i^k}{\rho_i}, \quad F_i^0 = 0$
- $BID_i^k = S_i^k + \frac{L_i^k}{\theta_i}$
- $V(t_2) - V(t_1) = \frac{r\rho_i(t_2 - t_1)}{\sum_{j \in B(t_1)} \rho_j}, \quad V(0) = 0$
- Packets are sent in the increasing order of their BIDs

Delay Bound for MPGPS-D

- Leaky bucket arrival with (σ_i, ρ_i)

- $\phi_i = \rho_i$

- $\sum_{j \epsilon Q} \phi_j = \sum_{j \epsilon Q} \theta_j < r$

- $\theta_i \geq \phi_i$:
$$d_i^k \leq \frac{\sigma_i - L_i^k}{\phi_i} + \frac{L_i^k}{\theta_i} + \frac{L^{max}}{r}$$

- $\theta_i < \phi_i$:
$$d_i^k \leq \frac{\sigma_i}{\phi_i} + \frac{L^{max}}{r}$$
if traffic arrival on other all the sessions during the time interval of $\frac{\sigma_i}{\phi_i}$
$$\sum_{j \epsilon Q \wedge j \neq i} A_j(t, t + \frac{\sigma_i}{\phi_i}) \leq \frac{\sigma_i}{\phi_i} \sum_{j \epsilon Q \wedge j \neq i} \phi_j$$

Comparison of Delay Bounds

- $\rho = \phi = 32\,Kbps$

L = 50 bytes		
Delay in PGPS (WFQ)	*25 ms*	
Delay in STFQ	*12.5 ms*	
θ in MPGPS-D	100	200
Delay in MPGPS-D	16.5 ms	14.5 ms
ϕ in PGPS	48 Kbps	55 Kbps

L = 100 bytes		
Delay in PGPS (WFQ)	*50 ms*	
Delay in STFQ	*25 ms*	
θ in MPGPS-D	100	200
Delay in MPGPS-D	33 ms	29 ms
ϕ in PGPS	48 Kbps	55 Kbps

Conclusions

- Overview of Packet scheduling Problem
- Delay Bounds achievable using Latency Rate server and Leaky Bucket Traffic
- Addressed the problem of Delay Bandwidth Coupling
- Computational Complexity of Scheduling algorithm