Packet Scheduling: End-to-End Delay Bounds

References

- Delay Guarantee of Virtual Clock server
  - Geoffrey G. Xie and Simon S. Lam, "Delay Guarantee of Virtual Clock Server," *IEEE/ACM Transactions on Networking*, Vol. 3, No. 6, December 1995.
- End-to-End Delay Guarantees and Bounds
  - Simon S. Lam and Geoffrey G. Xie, "Group Priority Scheduling," *IEEE/ACM Transactions on Networking*, Vol. 5, No. 2, April 1997.
  - Pawan Goyal, Simon S. Lam, and Harrick Vin, "Determining End-to-End Delay Bounds in Heterogeneous Networks," *ACM/Springer-Verlag Multimedia Systems*, Vol. 5, No. 3, May 1997.

Virtual Clock (VC) server - review

- Let r^f be the reserved rate in bits/s allocated to flow f. Let p denote a packet in flow f, with length $l(p)$ bits and arrival time, $A(p)$ (≥ 0)

- Each flow f has a "virtual clock", *priority*(f), which is zero initially and updated whenever a new packet in flow f arrives:

$$priority(f) \leftarrow \max\{priority(f), A(p)\} + \frac{l(p)}{r^f} \qquad (1)$$

The new value of *priority*(f) is assigned to packet p as its virtual clock value, denoted by $P(p)$

VC server –review (cont.)

- Whenever the VC server is ready for another packet, the packet among all flows with the smallest virtual clock value is selected
  - *FCFS within a flow*
  - non-preemptive

Meaning of virtual clock value

- Consider a hypothetical server with rate r^f dedicated to flow *f* (processor sharing). Let *F(p)* denote the finishing time of packet p. In this system,

$$F(p+1) = \max\{F(p), A(p+1)\} + \frac{l(p+1)}{r^f}$$

- Virtual clock value *P(p)* in previous slide is same as finishing time *F(p)* in processor sharing

Delay Guarantee of VC server

- VC by itself does not provide a *delay bound* in the usual sense, i.e., $L(p) - A(p)$ cannot be bounded by the server alone, where $L(p)$ denotes the departure time of packet p

- *Given admission control*, VC can provide a "delay guarantee" to a flow
  - with no assumption that sources are flow-controlled or well behaved
  - It is a conditional guarantee based upon a flow's reserved rate

Active flows

- VC server for a set F of flows, with reserved rate r^f allocated to flow f, for f in F

- Definition 1: A flow f is *active* at time t iff

$$priority\,(f) > t \qquad (2)$$

That is, a flow is active iff its virtual clock is *running faster* than real time.
- The virtual clock of flow f is driven by the flow's arrivals and packet lengths only [see eq. (1)]

"Active flow" interpretation

- At time *t*, the condition

$$priority(f) > t$$

holds at the hypothetical server (processor-sharing) of flow f if and only if *the hypothetical server is "backlogged"* i.e., there is at least one flow *f* packet waiting or being served

- During a busy period of the VC server (packet by packet),
  - flow *f* may be *active* when there is no flow *f* packet in the system (queue + server);
  - flow *f* may be *inactive* when the system has flow *f* packets

Capacity constraint condition

- Defintion 2: Let C denote the capacity, in bits/s, of a VC server. The server's capacity is not exceeded at time t iff the following condition holds:

$$\sum_{f \in a(t)} r^f \leq C \qquad (3)$$

where $a(t)$, a subset of F, is the set of flows that are active at time t

- **Above condition using $a(t)$ allows more packets to be admitted than using F**
  - but requires admission control for each packet arrival

Delay guarantee theorem

Theorem 1: If the capacity of a VC server has not been exceeded for a non-zero duration since the start of a busy period, then the following holds for every packet p served during the busy period:

$$L(p) \le P(p) + \frac{l_{\max}}{C} \qquad (4)$$

where $l_{\max}$ is the maximum packet size

(Proof by contradiction)

This bound is analogous to Parekh and Gallager's bound relating PGPS to GPS

Observations about Theorem 1

The theorem holds without any assumption that sources are well-behaved or flow-controlled.

- While each flow, say *f*, has a reserved rate r^f , its source can generate packets at a much larger rate than r^f . Packets can have arbitrary arrival times and packet lengths
- The delay guarantee in (4) holds even if the flows in *F* generate packets at an aggregate rate larger than *C* [assuming each flow is allocated its own buffers]
- It is a *conditional guarantee* based on the flow's reserved rate

VC server needs to enforce the capacity constraint

- The set of active flows, *a(t)*, changes dynamically. At any time the server can determine whether flow *f* is active by comparing *priority(f)* with the current time *t*. Thus the server can determine the set *a(t)*
- In theory, the server may perform *admission control upon the arrival of each packet* to prevent violation of its capacity constraint

Practical admission controls

Without a priori knowledge of source arrival characteristics:

- Flow sources are statically assigned reserved rates. Thus for all time *t*

$$\sum_{f \in a(t)} r^f \le \sum_{f \in F} r^f \le C$$

- On-demand assignment per session. A source can generate traffic only after it has been assigned a reserved rate on demand:

$$\sum_{f \in a(t)} r^f \le \sum_{f \in D(t)} r^f \le C$$

where D(t) is the set of sessions at time t with assigned reserved rates

Practical admission controls (more)

- On-demand assignment per application data unit (*burst*)
  - All packets of an application data unit are admitted or discarded together
    - Examples: (i) a picture in a video, (ii) a chunk of a video
  - Given a priori knowledge of source traffic statistics, a server can over-commit its capacity, making use of the fact that only some flows are active at any time
    - Statistical rather than deterministic guarantee

Properties of VC delay guarantee

- A delay guarantee of the form $L(p) \le P(p) + \beta$, where β is a constant, is a *conditional* guarantee
  - *A packet's delay is bounded from its expected arrival time based upon its reserved rate, that is, arriving early does not help*
  - On the other hand, packets arriving late do not get better service for not using their flow's reserved rate
- This *encourages on-time arrivals*, which are much better for buffer management in a switch (router)

Delay Bounds

- Consider a flow f with reserved rate r^f and packets indexed by n = 1, 2, … in order of arrival.
  - If $P(n)$-$A(n)$ is bounded above, then the delay of packet n, $L(n)$-$A(n)$, is bounded above
  - The goal of ***source control*** is to bound P(n) – A(n)
- **Method 1**: Ensure that the inter-arrival time of two consecutive packets is bounded below,

$$A(n+1) - A(n) \geq \frac{l(n)}{r^f} \qquad \Rightarrow \qquad P(n) = A(n) + \frac{l(n)}{r^f}$$

$$L(n) \leq P(n) + \frac{l_{\max}}{C} \qquad \text{Thus,}\ \ L(n) - A(n) \leq \frac{l(n)}{r^f} + \frac{l_{\max}}{C}$$

Leaky Bucket source control

- Let $Arrival_k^f(t_1, t_2)$ denote the number of flow *f* bits that arrive at server *k* in interval [t_1, t_2]
  - Bits of a packet arrive when the last bit of the packet arrives
  - The arrival function consists of a jump at each packet arrival time and is right continuous, $Arrival_k^f(t^-, t)$ is the size of the packet that arrives at time *t*
- Flow f conforms to Leaky Bucket with bucket size σ^f and rate ρ^f if and only if

$$\boxed{Arrival_k^f(t_1, t_2) \le \sigma^f + \rho^f (t_2 - t_1) \qquad \text{for} \quad 0 \le t_1 \le t_2}$$

Delay Bounds (cont.)

- **Method 2:** Source control by (σ, ρ) Leaky Bucket with
  bucket size σ and token arrival rate ρ *equal to*
  the reserved rate of the flow, r^f
- It is proved [Goyal, Lam, Vin 1997] that for all *n*

$$P(n) - A(n) \le \frac{\sigma}{\rho}$$

Thus, we have $L(n) - A(n) \le \dfrac{\sigma}{r^f} + \dfrac{l_{\max}}{C}$

For Spring semester 2017, skip ahead to slide 40

Generalizing Theorem 1

- The delay guarantee theorem holds as long as the server capacity is not exceeded by the sum of the reserved rates of flows that have had a packet served during the busy period.
- We can change the "active flow" definition (to admit more packets)

Definition 1′: A flow f is active at time t iff

(i) the system is not empty,

(ii) a flow f packet has arrived in the current busy period,

(iii) $priority(f) > t$

Generalizing Theorem 1 by another way

- We can achieve the generalization, *without changing the active flow definition*, by resetting the virtual clock of each flow to zero at the end of a busy period.
- Specifically, when the system becomes empty upon a packet departure, then for all f in F,

$$priority(f) \leftarrow 0$$

- *Side effect*: Resetting means that the "debt" of flow f is forgotten (forgiven)
  - Good or bad?

Deterministic End-to-End Delay Guarantees and Bounds

End-to-End Delay Guarantee

Consider a packet-switching network in which each packet is of variable, bounded size (in number of bits). Each communication channel in the network is statistically shared, and will be referred to as a *server*.

We will focus upon a flow, say f, which is a sequence of packets. Packets in the flow traverse a path of $K + 2$ nodes. Node 0 is the source of the flow, and node $K + 1$ is the destination. Nodes 1–K are servers. The network is to provide an end-to-end delay guarantee to the flow. Such a flow will be called a real-time flow. (We do not care whether or not the network also provides delay guarantees to other flows that statistically share the same servers.)

Notation for server k

β_k	A nonnegative time constant associated with node k (seconds).
$\tau_{k,k+1}$	Channel propagation time from node k to node $k+1$ (seconds) (each channel is assumed to be reliable and FIFO).
α_k	$= \beta_k + \tau_{k,k+1}$ (seconds).
$s_k^{\max}$	Maximum packet size at node k (bits).
C_k	Transmission rate of channel from node k to node $k+1$ (bits/second).
p	An arbitrary packet served (the packet may belong to any flow).

Guaranteed Deadline (GD) Server

- It provides the following service to flow f
  - packets in flow f depart in FIFO order
  - server ensures that the departure time of packet i is bounded as follows:

$$L_k^f(i) \leq P_k^f(i) + \beta_k \tag{1}$$

where the deadline on the right-hand side has two components: 1) a packet-dependent component $P_k^f(i)$ which depends on packet i (its arrival time, flow, size, priority, etc.) and 2) a nonnegative constant β_k.

The function $P_k^f(\cdot)$ is not yet specified. Any function may be used as long as, for a real-time flow, the server can ensure that, for all i, packet i departs by its deadline.

Many service disciplines in the literature belong to this class.

e.g., a Virtual Clock server

Traversing a path of GD servers

Consider a real-time flow f traversing the path from node 0 to node $K+1$. Nodes 1–K are GD servers (different service disciplines may be used at different nodes). The following lemma is immediate from the definition of $\boxed{\alpha_k = \beta_k + \tau_{k,k+1}}$

Lemma 1: For packet $i = 1, 2, \cdots$ in flow f and node $k = 1, 2, \cdots, K$, the arrival time of packet i at node $k+1$ is bounded as follows:

$$\boxed{A_{k+1}^{f}(i) \leq P_{k}^{f}(i) + \alpha_k.} \tag{2}$$

Notation for flow f

i, j	Indices of flow f's sequence of packets.
$\boxed{A_k^f(i)}$	Arrival time of packet i at node k (time when last bit of packet arrives).
$\boxed{P_k^f(i) + \beta_k}$	Deadline of packet i at node k where β_k is defined above and $P_k^f(i)$ is a packet-dependent component.
$\boxed{L_k^f(i)}$	Departure time of packet i at node k (time when last bit of packet leaves).
$\boxed{s^f(i)}$	Size of packet i (bits).

$A_k^f(i), P_k^f(i)$, and $L_k^f(i)$, for $i \geq 1$, are positive real numbers. Indices i and j are positive integers. Additionally, we also use m, n, and l as positive integers whose meanings depend upon context.

Reference clock values for flow f

$v^f(i)$ a time constant associated with packet *i* (seconds)

$V_k^f(i)$ reference clock value of packet *i* at server *k*, to be interpreted as ***expected finishing time*** of packet *i* at server *k*

$V_k^f(0)$ is 0 by definition and for $i \geq 1$

$$V_k^f(i) = \max\{V_k^f(i-1), A_k^f(i)\} + v^f(i) \qquad (3)$$

- For the special case of VC servers

$v^f(i) = s^f(i) / \lambda^f(i)$ where $\lambda^f(i)$ is reserved rate for packet i in flow f

The virtual clock value of packet *i* at server *k* is its reference clock value and $P_k^f(i) = V_k^f(i)$ for all *i*

Important relations

- Each server k ensures that packet *i* in flow f departs by its deadline which is $P_k^f(i) + \beta_k$

- $A_{k+1}^f(i)$ depends on $P_k^f(i)$ as shown in (2)

  $V_k^f(i)$ depends on $\mathrm{A}_k^f(i)$ as shown in (3)

- For a VC server, *k*, we know that

$$P_k^f(i) = V_k^f(i)$$

What about other scheduling disciplines?

Putting the relations together

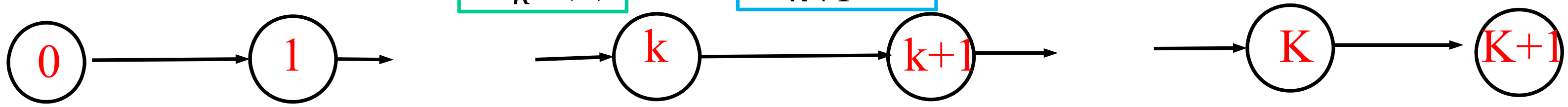
Consider packet i in flow f. *Its end to end delay is*

$$\boxed{A^f_{K+1}(i) - A^f_1(i)}$$

$$\boxed{A^f_1(i) \geq V^f_1(i) - \frac{\sigma}{\rho}}$$

<- source control

$A^f_1(i)$ $\boxed{A^f_k(i)}$ $\boxed{A^f_{k+1}(i)}$ $A^f_K(i)$ $A^f_{K+1}(i)$



source $L^f_1(i)$ $\boxed{L^f_k(i)}$ $L^f_{k+1}(i)$ $L^f_K(i)$ destination

$$\boxed{\leq P^f_k(i) + \beta_k}$$

<- deadline guarantee by server k

β_k is delay due to non-preemption

$$\boxed{A^f_{k+1}(i) = L^f_k(i) + \tau_{k,k+1} \leq P^f_k(i) + \alpha_k}$$

Difference between a packet's deadline and reference clock value

For each GD server, suppose we know, for all i ,

$$P_k^f(i) - V_k^f(i)$$

Lemma 2: For packet $i = 1, 2, \cdots$ in flow f and node $k = 1, 2, \cdots, K-1$,

$$V_{k+1}^f(i) \leq V_k^f(i) + \max_{1 \leq j \leq i} \{v^f(j) + (P_k^f(j) - V_k^f(j))\} + \alpha_k. \tag{4}$$

Proof in [Lam and Xie 1997] by induction

Recall that $v^f(j) = \dfrac{s^f(j)}{\lambda^f(j)}$ and

$$\alpha_k = \beta_k + \alpha_{k,k+1}$$

End-to-end delay guarantee theorem

Theorem 1: For packet $i = 1, 2, \cdots$ in flow f, the arrival time of packet i at the destination is bounded as follows:

$$A_{K+1}^f(i) \le V_1^f(i) + \sum_{k=1}^{K-1} \max_{1 \le j \le i} \{ v^f(j) + (P_k^f(j) - V_k^f(j)) \} + (P_K^f(i) - V_K^f(i)) + \sum_{k=1}^{K} \alpha_k. \quad (5)$$

Proof in [Lam and Xie 1997] by applying Lemma 2 recursively

End-to-end delay upper bound

- The end-to-end delay of packet *j* is $A_{K+1}^{\ f}(j) - A_1^{\ f}(j)$
- The end-to-end guarantee in (5) provides an upper bound on the end-to-end delay if
  - a source control mechanism is used such that $V_1^{\ f}(j) - A_1^{\ f}(j)$ has a finite upper bound,

  and
  - server *k* is a GD server, *1 ≤ k ≤ K*, such that the term $P_k^{\ f}(j) - V_k^{\ f}(j)$ has a finite upper bound

Example of source control

If the source of flow f is controlled by a leaky bucket with bucket depth σ and token rate $\rho = \lambda^f$, which is the reserved rate of flow f, then for all packet i in the flow [Goyal, Lam, Vin 1997]

$$\boxed{V_1^f(i) \leq A_1^f(i) + \frac{\sigma}{\rho}} \qquad (6)$$

To obtain an end-to-end delay upper bound for flow f, $V_1^f(i)$ is instantiated to $A_1^f(i) + \sigma/\rho$ in the delay guarantee formula of Theorem 1.

End-to-end delay upper bound (cont.)

- Note that different service disciplines may be chosen for different servers along a path
  - also, the term $P_k^f(j) - V_k^f(j)$ may be positive or negative
- With Theorem 1, the problem of providing an upper bound on the end-to-end delay for a flow along a path of heterogenous servers is **reduced to a set of single-node problems**!

Examples of GD servers

- The GD class of servers is general including many packet schedulers in the literature. They differ in their $P_k^f(.)$ functions, β_k constants, and admission control conditions
- We will show four:
  - Virtual Clock
  - WFQ/PGPS
  - Delay-EDD
  - Leave-in-Time

1. Virtual Clock (VC) as a GD server

For a VC server, the P values are virtual clock values computed as follows [15] for all $j \geq 1$:

$$\boxed{P_k^f(j) = \max\{P_k^f(j-1), A_k^f(j)\} + v^f(j)} \qquad (7)$$

where $P_k^f(0) = 0$ and $\boxed{v^f(j) \text{ is equal to } s^f(j)/\lambda^f(j)}$. Under certain admission control conditions, the VC server provides the guaranteed deadline in (1) with $\boxed{\beta_k = s_k^{\max}/C_k}$ [11]. From (3) and (7), it is trivial to show that, for all j,

$$\boxed{P_k^f(j) - V_k^f(j) = 0.} \qquad (8)$$

2. WFQ/PGPS as a GD server

The packet deadline at server *k* is, using $P_k^f(j) = L_{k,GPS}^f(j)$,

$$\boxed{L_{k,PGPS}^f(j) \le L_{k,GPS}^f(j) + \frac{s_k^{\max}}{C_k}}$$ from Parekh and Gallager [1993]

where $L_{k,GPS}^f(j)$ is the departure time of k being a GPS server with weights $\{\mathrm{w}_g,\ g \in F\}$

At GPS server k, consider the rate, $$\boxed{\lambda^f \le \frac{w_k^f C_k}{\sum_{g \in b_k(t)} w_k^g}}$$

where $b_k(t)$ is the set of backlogged flows, to be the reserved rate for flow f

Set of flows must be finite for the reserved rate to be nonzero for all t

2. WFQ/PGPS as a GD server (cont.)

From Goyal, Lam, and Vin [1997] eq. (7)

$$\boxed{L^f_{k,GPS}(j) \le \max\{L^f_{k,GPS}(j-1), A^f_k(j)\} + \frac{s^f(j)}{\lambda^f} \quad .}$$

$$V^f_k(j) = \max\{V^f_k(j-1), A^f_k(j)\} + v^f(\mathrm{j}) \qquad \text{from eq. (3)}$$

$$\text{where } v^f(j) = \frac{s^f(j)}{\lambda^f}.$$

$$\boxed{\text{Therefore, } P^f_k(j) = L^f_{k,GPS}(j) \le V^f_k(j)}$$

$$\boxed{\text{Finally, } P^f_k(j) - V^f_k(j) \le 0}$$

3. Delay-EDD as a GD server

For a Delay-EDD server [2], the P values of packets are computed as follows [14] for all $j \geq 1$:

$$\boxed{P_k^f(j) = \max\{A_k^f(j) + d_k^f, P_k^f(j-1) + v^f\}} \qquad (10)$$

where $P_k^f(0) = -v^f$, $\boxed{d_k^f \text{ is a local delay bound}}$ for every packet in flow f, and $v^f(j) = v^f = s^f/\lambda^f$, with s^f and λ^f being the same for all j. If certain $\boxed{\text{schedulability conditions}}$ are met, then a Delay-EDD server provides the guaranteed deadline in (1) with $\boxed{\beta_k = 0}$ [2]. By induction, it is easy to show that, for all $j \geq 1$,

$$\boxed{P_k^f(j) - V_k^f(j) = d_k^f - v^f}. \qquad > \mathbf{0} \qquad (11)$$

2/7/2017

4. Leave-in-Time as a GD server

For a leave-in-time server [3], the P values of packets are computed as follows[5] for $j \geq 1$:

$$\boxed{P_k^f(j) = \max\{A_k^f(j), V_k^f(j-1)\} + d_k^f(j)} \qquad (12)$$

$$\boxed{V_k^f(j) = \max\{A_k^f(j), V_k^f(j-1)\} + v^f(j)} \qquad (13)$$

where $V_k^f(0) = 0$, $\boxed{d_k^f(j) \text{ is the local delay bound}}$ of packet j, and $v^f(j) = s^f(j)/\lambda^f$, with the reserved rate λ^f the same for every packet in flow f. Under certain admission control conditions, a leave-in-time server provides the guaranteed deadline in (1) with $\boxed{\beta_k = s_k^{\max}/C_k}$ [3]. Subtracting (13) from (12), we have for all j

$$\boxed{P_k^f(j) - V_k^f(j) = d_k^f(j) - v^f(j)} \quad > 0 \qquad (14)$$

End-to-end delay bound for a path of servers that use VC or WFQ/PGPS scheduling

Using Eq. (5) of Theorem 1 together with leaky bucket source control, we get

$$A_{K+1}^{f}(i) - A_{1}^{f}(i) \leq \frac{\sigma^{f}}{\rho^{f}} + (K-1)\max_{1\leq j\leq i} \frac{s^{f}(j)}{\rho^{f}} + \sum_{k=1}^{K} \alpha_k$$

$$= \boxed{\frac{\sigma^{f} + (K-1)\max_{1\leq j\leq i} s^{f}(j)}{\rho^{f}} + \sum_{k=1}^{K} \alpha_k}$$

A tight bound

where $\rho^{f} \leq \lambda^{f}$ and $\alpha_k = \beta_k + \tau_{k,k+1}$, where $\beta_k = \dfrac{s_{\max}^{k}}{C_k}$

The end-to-end delay bound of Parekh and Gallager

A. Parekh and R. Gallager, "A generalized processor-sharing approach to flow control in integrated services networks: the multiple node case," *IEEE/ACM Trans Networking*, April 1994, equation (39).

Previously known bound for rate-proportional processor sharing (RPPS) rate assignment of PGPS networks when the sources conform to Leaky Bucket is

$$\frac{\sigma^f + \boxed{2}(K-1)\times \boxed{s_{\max}^f}}{\rho^f} + \sum_{k=1}^{K} \alpha_k$$

a loose bound

where $s_{\max}^f$ is the maximum packet size for flow f

Summary

- End-to-end delay bounds can be provided to flows by a variety of GD servers
  - Delay guarantee theorem
  - Decomposition of the end-to-end delay bound problem into single-node and source control problems

- Both VC and WFQ/PGPS servers provide the best delay bound (a tight bound)
  - VC is easier to implement
  - WFQ is slightly more fair

The end