Network QoS
371-2-0213

Lecture 6

Gabriel Scalosub

Outline

Model

- Commitments model we saw: managing a single AF PHB
- Now: a more general DiffServ Model
    - Managing multiple AF PHBs / EF classes
    - Differentiate between packets corresponding to different classes
    - E.g.: EF "$>$" AF41 "$>$" AF43
        - scheduling: link access (delay / rate)
        - buffer management: drop rate
- System model: slotted time
- Traffic model
    - uniform size packets (WLOG, size is 1)
    - for each packet p we have
        - $a(p)$: arrival time
        - $w(p)$: packet *weight*
        - $s_A(p)$: forwarding/delivery time by algorithm A
- Packet weight gives means for differentiation
    - can be considered as part of PHB

Model

- Queue model
  - Single FIFO queue Q
    - buffer of size B
    - service rate r
    - preemptive: may drop enqueued packets

Goal

Maximize overall weight of forwarded packets

- What algorithm would you pick?
- Assumptions and notation:
  - $\mathcal{B}_A(t)$: set of packets in the buffer of algorithm A at time t
  - $S_A(t)$: set of packets sent by algorithm A at time t
  - $w(P)$: overall weight of set of packets P
    - i.e., $w(P) = \sum_{p \in P} w(p)$
  - for simplicity, assume $r = 1$

The Greedy Algorithm

Algorithm 1 G: upon the arrival of p at time t

if $|\mathcal{B}_{\mathrm{G}}(t)| < B$ **then**
    accept p
else
    $q \leftarrow \arg\min \{w(q') \mid q' \in \mathcal{B}_{\mathrm{G}}(t)\}$
    if $w(p) \leq w(q)$ **then**
        drop p
    else
        drop q
        accept p
    end if
end if

- Notation:
  - $A(t)$: packets arriving at t
  - $D_{\mathrm{G}}(t)$: packets dropped by G at t

Competitive Analysis of Greedy - Upper Bound

- $V(t)$: B highest weight packets arriving during $[t+1, t+B]$

Lemma

For any time t,
$w(\mathcal{B}_{\mathrm{G}}(t)) + w(V(t)) \leq \sum_{i=1}^{B} w(s_{\mathrm{G}}(t+i)) + w(\mathcal{B}_{\mathrm{G}}(t+B))$

- LHS: current buffer at t + best potential arrivals by $(t+B)$
- RHS: actual deliveries by $(t+B)$ + future buffer at $(t+B)$

Competitive Analysis of Greedy - Upper Bound

- $V(t)$: B highest weight packets arriving during $[t+1, t+B]$

Lemma

For any time t,
$w(\mathcal{B}_{\mathrm{G}}(t)) + w(V(t)) \leq \sum_{i=1}^{B} w(s_{\mathrm{G}}(t+i)) + w(\mathcal{B}_{\mathrm{G}}(t+B))$

Proof.

- Y_0^t: $\mathcal{B}_{\mathrm{G}}(t) \cup V(t)$
- $Y_i^t = Y_{i-1}^t \cup A(t+i) \setminus D_{\mathrm{G}}(t+i)$ (for $i \geq 1$)
  - By definition of G, $w(Y_{i-1}^t) \leq w(Y_i^t)$

    $\Rightarrow$ $w(Y_0^t) \leq w(Y_B^t)$
  - $Y_B^t \subseteq \mathcal{B}_{\mathrm{G}}(t) \cup \bigcup_{i=1}^{B} A(t+i)$
  - none of the packets in Y_B^t are discarded by $t+B$
  - packets in Y_B^t either sent by $t+B$, or in $\mathcal{B}_{\mathrm{G}}(t+B)$ □

Competitive Analysis of Greedy - Upper Bound

Theorem

Algorithm G *is 2-competitive*

Proof.

- Partition time into intervals of length B
  - $I_k = [kB, (k+1)B),\ k = 0, 1, \ldots$
- $U(k)$: set of packets accepted by $\textsc{Opt}$ during I_k
- $|I_k| = B$
  - $|U(k)| \leq 2B$
  - $w(U(k)) \leq 2w(V(kB))$

Competitive Analysis of Greedy - Upper Bound

Theorem

Algorithm G *is 2-competitive*

Proof.

- Previous lemma:

$$w(V(kB)) \leq \sum_{i=1}^{B} w(s_{\mathrm{G}}(kB+i)) + w(\mathcal{B}_{\mathrm{G}}(kB+B)) - w(\mathcal{B}_{\mathrm{G}}(kB))$$

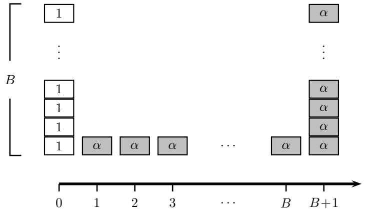
- for $T = \max\{a(p)\} + B$

$$\sum_k w(V(kB)) \leq \sum_t w(s_{\mathrm{G}}(t)) + \underbrace{w(\mathcal{B}_{\mathrm{G}}(T))}_{0} - \underbrace{w(\mathcal{B}_{\mathrm{G}}(0))}_{0}$$

$\Rightarrow$ $w(OPT) \leq 2\sum_k w(V(kB)) \leq 2w(\mathrm{G})$ $\square$

Competitive Analysis of Greedy

- The analysis of G is essentially tight
  - Consider the traffic:



    - $w(\mathrm{G}) = B(1+\alpha)$
    - $w(\textsc{Opt}) = 2B\alpha$
  - the ratio is $\frac{2B\alpha}{B(1+\alpha)} = \frac{2\alpha}{1+\alpha} = 2 - \frac{2}{1+\alpha}$
  - tends to 2 as α goes to infinity
- Uses only 2 distinct values

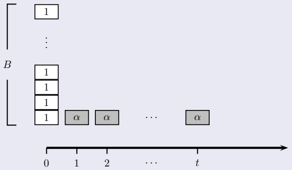
Competitive Analysis - Deterministic Lower Bound

Theorem

No deterministic algorithm has competitive ratio better than 1.281

Proof.

- Fix some deterministic algorithm $\textsc{Alg}$
- Consider traffic similar to previous
  - continues until earliest time $t \leq B$:
    - $\textsc{Alg}$ sends an α-packet at $(t+1)$, or
    - $t = B$
- Scenario one:
  - sequence ends
    - $\textsc{Alg}$ gains $t + t\alpha$
    - $\textsc{Opt}$ would have gained $B + t\alpha$



Competitive Analysis - Deterministic Lower Bound

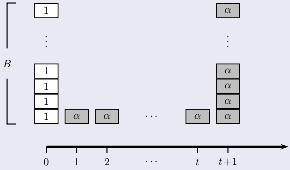
Theorem

No deterministic algorithm has competitive ratio better than 1.281

Proof.

- Scenario two:
  - sequence continues similar to greedy's tight example
    - $\textsc{Alg}$ gains $t + B\alpha$
    - $\textsc{Opt}$ would have gained $(t + B)\alpha$
- Competitive ratio of $\textsc{Alg}$ is at least

$$\max\left\{\frac{B + t\alpha}{t + t\alpha}, \frac{(t + B)\alpha}{t + B\alpha}\right\}$$



Competitive Analysis - Deterministic Lower Bound

Theorem

No deterministic algorithm has competitive ratio better than 1.281

Proof.

- The minimum (over t) is obtained when equality holds
  - Solving (for t): $t_0(B,\alpha) = B\frac{\sqrt{(\alpha-1)^2+4\alpha^3}-\alpha+1}{2\alpha^2}$
  - define $t_0^*(\alpha) = t_0(B,\alpha)/B$
  - competitive ratio is at least $\frac{1+\alpha t_0^*(\alpha)}{(1+\alpha)t_0^*(\alpha)}$
- solving (numerically, for α), this is maximized for $\alpha \approx 4.01545$
- Competitive ratio of $\textsc{Alg}$ cannot be better than 1.281

$\square$

Better Algorithms: What Would You Do?

- Recall Greedy:
- Improvement options:
  - consider *complex p-q relationships*
    - drop q only if $w(q) \ll w(p)$
    - not just comparison-based – actual values matter!!
  - consider the overall *present "state"* of the buffer
    - how much value is currently in the buffer?
    - do we currently have enough to "cover" for not preempting?
    - proactive dropping? (avoid delaying others...)
  - consider the present state and *history*
    - how much value did we gain recently?
    - did we gain enough to "cover" for not preempting?
  - *admission control*
    - drop even if the buffer isn't full? (proactive...)
    - may depend on / be combined with all of the above
  - *toss coins...*

Algorithm 1 G: upon the arrival of p at time t

```
if |B_G(t)| < B then
    accept p
else
    q ← arg min {w(q') | q' ∈ B_G(t)}
    if w(p) ≤ w(q) then
        drop p
    else
        drop q
        accept p
    end if
end if
```

How Was This Useful?

- Detailed results we saw are due to:
  - upper bound: Mansour, Patt-Shamir, Lapid (2000)
  - lower bound: Kesselman, Lotker, Mansour, Patt-Shamir, Schieber, Sviridenko (2001)
- Many papers tried to close the gap
  - many algorithms!!
- Most of the focus: 2-valued case
  - matching upper bound finally by Englert&Westermann (2008)
- What about randomized algorithms?
  - strictly better than deterministic...

deterministic		randomized	
upper	lower	upper	lower
	1.282		
$2 - \frac{2}{\alpha+1}$			
1.894			
1.544			
1.304			
		1.25	1.197
1.282			

Non-preemptive, 2-valued FIFO

- Actually, the first model considering packet weights
    - Aiello, Mansour, Rajagopolan, Rosén (2000)
- Focus solely on admission control
- greedy is bad: tight $1/\alpha$-competitive
    - upper bound: every 1-packet mapped to an α-packet
    - lower bound: non-preemption implies algorithm may only accept 1-packets
- lower bound: $\frac{\alpha}{2\alpha-1}$
- matching upper bound by Andelman, Mansour, Zhu (2003)

General Values

- Many more papers...

deterministic		randomized	
upper	lower	upper	lower
4			
2			
	1.282		
	1.414		
	1.419		
	1.434		
1.983			
1.75			
			1.25
		1.75	
1.732			

More Models

- Loss-bounded model
  - measure: packets dropped (instead of delivered)
  - Kesselman&Mansour (2003)
  - constant competitive
    - also implies good competitive ratio in "standard" model
- Multiple Queues
  - multiple output-queues with shared memory
    - unit-values (not really "DiffServ")
    - goal is doing load-balancing (e.g., longest-queue-drop)
    - constant competitive
  - multiple input-queues
    - essentially a scheduling problem: which IQ to schedule
    - goal is doing load-balancing (e.g., longest-queue-first)
    - mostly constant competitive
- Many more...