Lec24-mwf

Sunday, March 20, 2011 11:04 AM

REM

Wednesday, March 25, 2009 9:36 AM

Ⓠ Is this algorithm as simple as TCP?

Ⓐ Yes: source only needs to know prices
link only needs to the aggregate rate

No: need to update & communicate "prices"
No AIMD.

It turns out that the "prices" are closely related to the queue-length

- Let $Q_l(t) = q_l(t)/\gamma$

then

$$Q_l(t+1) = [Q_l(t) + \left(\sum_s H_l^s x_s(t) - R_l) \right)^+$$

- This is simply the queue-evolution equation
∋ "price" is simply a scaled version of the queue length!

Ⓠ How to communicate the queue-length?

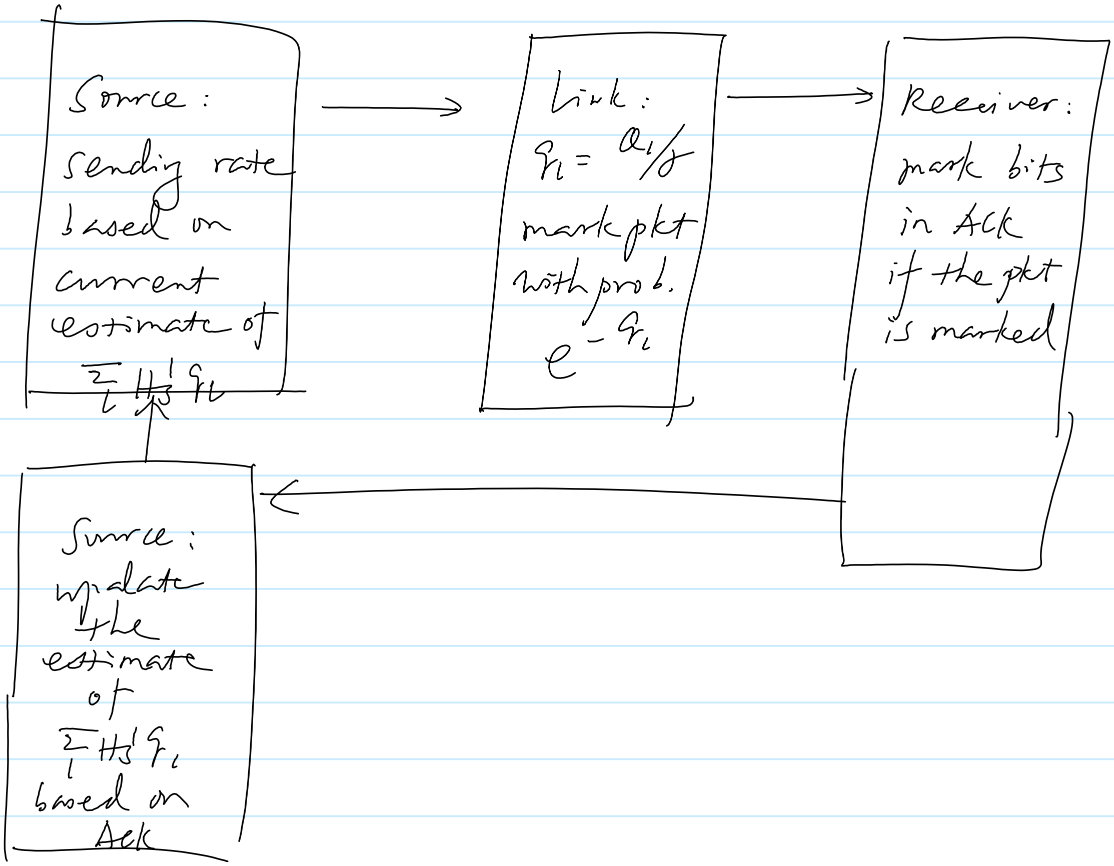
Ⓐ
- by explicit control messages
- by packet drops or packet marks

<u>Random</u> - Exponential Marking (REM)

- Each packet is marked at link l with probability $1-e^{-q_l}$
- The probability that a packet is marked after it passes through the path is

$$1-e^{-\sum_l H^l_s q_l}$$

- Hence, by counting the fraction of packets marked, the source can estimate $\sum_l H^l_s q_l$.



Potential Issues
- When q_l is conveyed by these markings there will be significant noise in their values
  $\Rightarrow$ Will the dual algo still converge to the optimal solution?

⇒ Will the "dual" algo still converge to the optimal solution?

- When the source adjusts its rate $x_s(t)$, there may be a delay when the link can observe the new packet rates.
  - ⇒ Will convergence still happen under communication delay?
- We will study these issues later.

- When q_l is conveyed by these markings, there will be significant noise in the values estimated at the source
- Thus, setting x_s directly by maximizing

$$U_s(x_s) - x_s \sum_l q_l H_s^l$$

may be cause unnecessary fluctuation

- Instead, the source may also update x_s via gradient-descent

$$\dot{x}_s = K_s \left[U_s'(x_s(t)) - \sum_l H_s^l q_l(t) \right]$$

- As we discussed earlier, this can be converted into AIMD window-based control
- Combined with

$$\dot{q}_l = \beta_s \left[\sum_s H_s^l x_s(t) - R^l \right]$$

This becomes a "primal-dual" controller.

- Equations like these can be used to model the dynamics of TCP (at the source) plus more sophisticated dropping/marking mechanism (at the links, e.g. REM, RED).

Critique of dual controller

Thursday, March 19, 2009 5:22 PM

Pros:

- Simple to implement
  - only feedback is queue-length (or packet loss)
  - source does not need to know other users' information
  - links do not need to maintain per-user information.
- Produce an exact solution to the congestion control problem.
  - At equilibrium, aggregate rate $\le$ link capacity.

    $\Rightarrow$ no dropping
- Queue-length evolution is explicitly accounted for.

Cons:

- At equilibrium, there will be queue backlog

  ⇒ packet delay increases if the queue-length increases.

(20)

AQM

Friday, March 20, 2009 3:12 PM

- Can we eliminate the coupling between queue and congestion price?

Virtual Queues

- A virtual queue has a capacity that is slightly below the true capacity of the link.

- It does not really store packets. Only needs to maintain its length

- By dropping/marking packets according to the virtual queue length, a link can provide advanced congestion indication.

Virtual Queue for Primal Controllers

- Recall in primal solutions, at equilibrium the offered load at each link will be greater than the capacity

$$\Rightarrow \text{inevitable packet loss}$$

- Using virtual queue

$$\dot{x}_s = K_s \Big[U_s'(x_s) - \sum_l H_s^l p_l \big(\Sigma H_s^l x_s , \hat{R}_l \big) \Big)$$

If the virtual capacity $\hat{R}_l$ is set correctly, it is possible to achieve, at equilibrium

$$\sum_s H_s^l x_s = R^l$$ ↑ true capacity.

- One possibility is to "slowly" update $\hat{R}^l$ by

$$\dot{\hat{R}}_l = -\left(\sum H_s^l x_s - R^l\right)$$

- Called AVQ (Adaptive Virtual Queue) in the literature.

Virtual Queue for dual controller

- Recall that in dual controller, at equilibrium there will be considerable amount of queue backlog

$$Q_l(t) = \frac{q_l(t)}{\gamma}$$

- More congestion $\Rightarrow q_l(t) \uparrow \Rightarrow$ delay ↑

- Using virtual queues, we can decouple prices from the real queue-length

- Prices are updated according to an virtual queue

$$\dot{q}_l = \beta\left(\sum_s H_s^l x_s(t) - (1-\varepsilon) R_l\right)$$

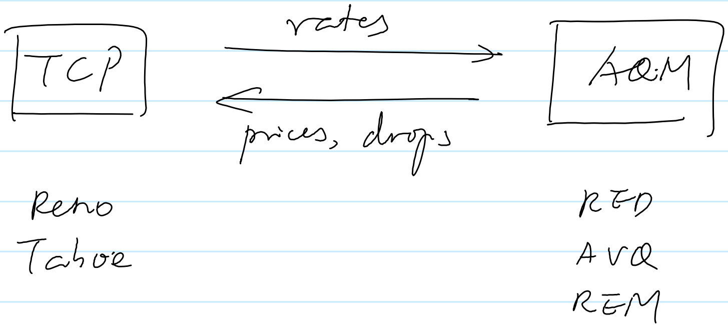
- At equilibrium, $\sum_s H_s^l x_s = (1-\varepsilon) R_l$

$\Rightarrow$ The real queue (at capacity R_l) will see very small backlog.

- These are examples of AQM (Active Queue Management).

(Active "Queue" Management).

- Each link "actively" manages the content of the queue, and notifies the source by "active" dropping/marking
- Includes REM, RED



- Key: simple to implement.
  - For example, the router does not need to maintain per-flow information

---

At some high-level, the dynamics of the system can be written as a set of difference or differential equations. We can then ask

- Do they solve the congestion control problem (the utility max. problem)?
- Do they have low delay, low packet loss?
- Are they robust to delayed feedback and/or estimation errors?

① "End-to-end Congestion Control Schemes: Utility
Functions, Random Losses and ECN Marks,"
S. Kunniyur and R. Srikant,
IEEE/ACM Transactions on Networking,
Vol. 11, No. 5, October 2003

② S. H. Low and D. E. Lapsley, "Optimization Flow Control-I: Basic Algorithm and Convergence," IEEE/ACM Transactions on Networking, vol. 7, no. 6, pp. 861-874, December 1999.

㊿

Cross-layer formulation

Sunday, February 01, 2009 12:25 PM

In an optimization approach, it is not difficult to incorporate controls at multiple layers into a unified optimization problem.

- Physical layer:
  - power control, water-filling
  - uses rate-power function
- MAC:
  - scheduling
- Network Layer:
  - multi-path routing
  - node-balance equation
- Transport layer
  - utility maximization
  - revenue maximization

So we have various combinations.
Key consideration is
- convexity
- distributed/decomposed solution.

One way of putting all together

$$\max \sum_s U_s\Big(\sum_j x_{sj}\Big)$$ - utility / congestion control

sub to

$H_{s_j}^l = 1$ if user s uses link l on path j →

$$\sum_s \sum_j H_{s_j}^l x_{s_j} \le r_l$$

$$r_l \le \sum_{k=1}^{K} \theta_k r_l^k, \forall l$$ – k channels

↑ prob. of channel state k

dependency on the the channel condition at state k.

may not be convex →

$$r_l^k \le g_l(\vec{p}^k, s^k)$$ – power control adaptive coding/ modulation

rate-power function

power assignment for schedule k.

Other formulations

- replace $H_{s_j}^l$ by node-balance eqn

Ref: Lin, Shroff & Srikant, "A Tutorial on Cross-Layer Optimization in Wireless Networks," IEEE Journal on Selected Areas in Comm, Special Issue on "Non-Linear Optimization of Comm Systems," 2006

- In general, not a convex problem.
    - The function $g_i(\cdot)$

(20)

Convexifying the problem through time-sharing

Friday, March 27, 2009 9:30 AM

- For simplicity, assume no channel variation first
  - $k = 1$
- In general, let p_i be the control of link i
  - it could be the power allocation $p_i \in [0, p_{i,max}]$
  - or could be an indicator of whether the link is activated or not $p_i \in \{0, 1\}$
- For each $\vec{p} = (p_1, p_2, \cdots p_L)$, there is a corresponding $\vec{r} = g(\vec{p})$.
  - rate-power function
  - $\vec{r} = (r_1, r_2, \cdots, r_L)$
- Without time-sharing, the capacity-region is

$$\{\vec{r} \mid \vec{r} = g(\vec{p}) ; p \in \Pi\}$$

where Π is the set of all feasible control vectors

  - typically non-convex

- Time-sharing: let us use one control $\vec{p_1}$ for θ fraction of the time, and use $\vec{p_2}$ for $1-\theta$

fraction of the time, and use $\vec{p}_2$ for $1-\theta$ fraction of the time

- The new rate vector will be
  $$\theta \cdot \vec{r}_1 + (1-\theta)\vec{r}_2$$
  where
  $$\vec{r}_1 = g(\vec{p}_1) \quad , \quad \vec{r}_2 = g(\vec{p}_2)$$

- This is exactly the convex combination of $\vec{r}_1$ & $\vec{r}_2$!

- With time-sharing, the capacity-region becomes

$$\mathrm{Conv}\{\vec{r} \mid \vec{r} = g(\vec{p});\ p \in \Pi\}$$

which is by definition a convex set.

With Channel-variations

Skip

- The system can be in one of M channel states $S_1, \cdots, S_M$

- θ_K = prob. that the system is in state S_K.

- rate-power function becomes

$$\vec{r} = g(\vec{p}, S_K)$$

- The capacity region (with time-sharing) becomes

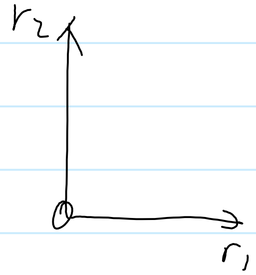
$$\sum_{k=1}^{M} \theta_k \; \mathrm{Conv}\{\vec{r} \mid \vec{r} = g(\vec{p}, S_k), \vec{p} \in \Pi\}$$

Example:

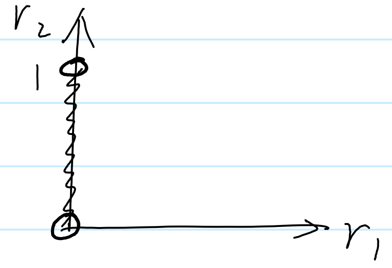
- One BS
- Two users
- BS can only transmit to one user at a time
- Four states

$$(0,0) \qquad (0,1) \qquad (\underset{\substack{\uparrow \\ \text{ON} \\ \text{for} \\ \text{user 1}}}{1}, \underset{\substack{\uparrow \\ \text{OFF} \\ \text{for} \\ \text{user 2}}}{0}) \qquad (1,1)$$

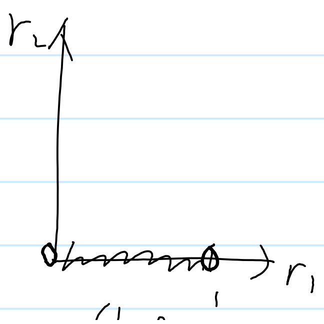
- The convex-hull for each state

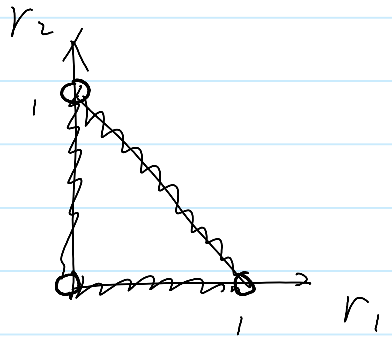


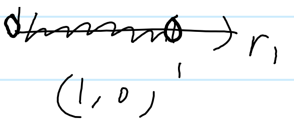
(0, 0)

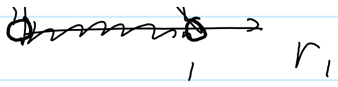


(0, 1)

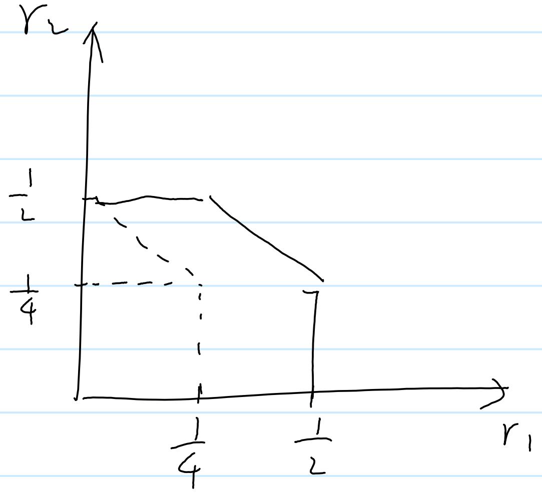








- Suppose $p_{00} = p_{01} = p_{10} = p_{11} = \frac{1}{4}$.
- What does the overall capacity region looks like.



- With such convexification, our cross-layer optimization problem becomes convex!
- Replace

$$r_i^k \le g_i(\vec{p}^k, s^k)$$

by

$$(r^k) \in \text{Conv}\{\vec{r} \mid \vec{r} = g(\vec{p}, s_k), \vec{p} \in \Pi\}$$

㉕