Lec17

Sunday, February 23, 2020 10:02 AM

HW3 is on the web.

Need for Cross-layer Design - 10min

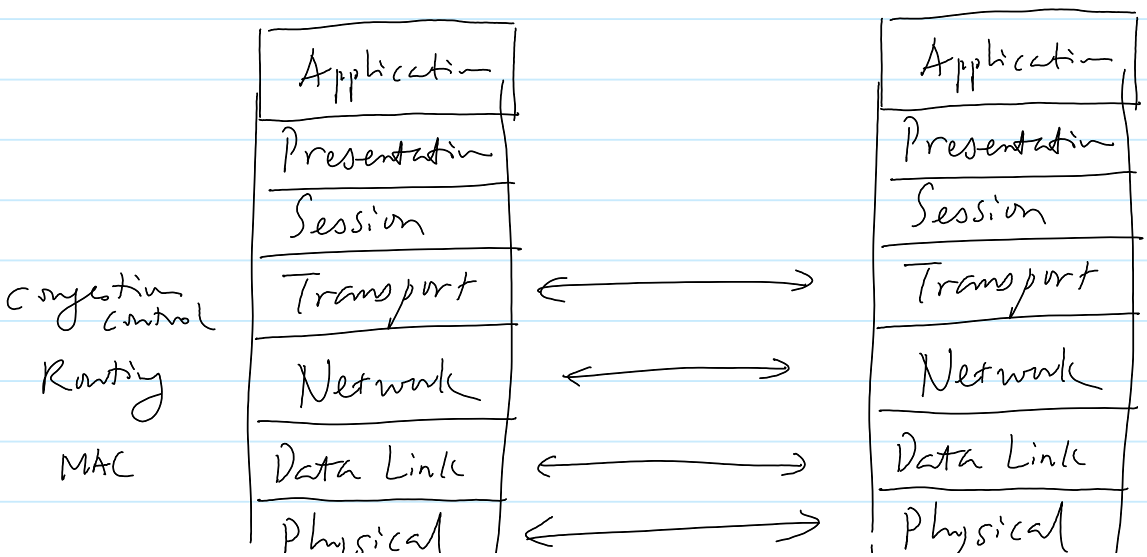
Saturday, January 31, 2009 5:37 PM

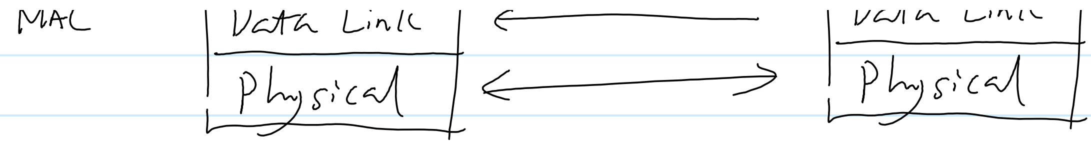
We have seen an example of joint congestion control & multipath routing.

This is just one example of "cross-layer control"

Q Why do we want to consider multiple layers together?

- In Wireline networks, often the protocols are classified into layers.
- Layering is a form of hierarchical modularity.
- The higher layer uses the service provided by the lower layer. But it does not need to know the inner working of the lower layer



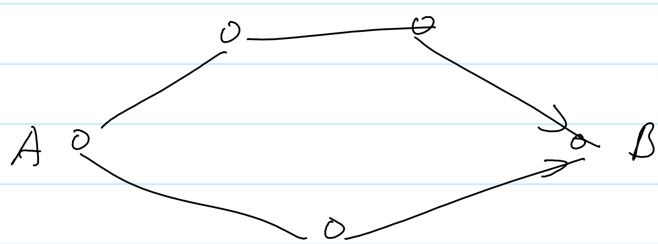


- Benefits of Modularity.
    - easy to understand
    - easy to change.

- However, for wireless networks, examples have been found where such a layering architecture can limit performance.

Example.

- Typically, routing is designed to minimize the # of hops



- Tend to use "long" links.

- In wireless networks, long transmission link can suffer from a low SINR

$\Rightarrow$ poor end-to-end performance.

- It would be better if the routing protocol takes into account the physical-layer characteristics.

- Pitfalls of Cross-layer Design
  - loss of modularity
  - fragile solution that is hard to change.

- Nonetheless, the Lyapunov-based approach allows us to easily integrate different functionality together

  - Many of the solutions consist of a MAC component of the form

$$\max_{\substack{\vec{P} \in \Theta \\ \vec{r} = g(\vec{P}, K(t))}} \sum_l q^l(t) r_l$$

- The queue-length becomes the "glue" across layers.

- We will not study all of these problems. Instead, focus on routing.

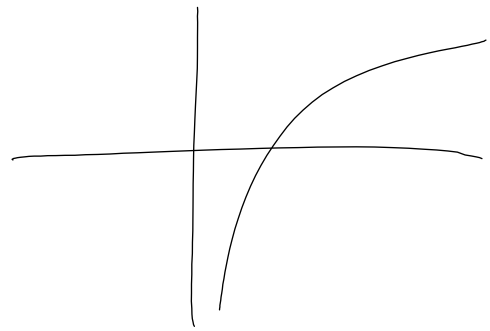
The project 1 deals with one flavor of rate-control & fairness.

Joint congestion control - 5min

Sunday, February 11, 2018 9:48 AM

- Suppose that we want to also choose the arrival rate X_s each flow s to maximize some system objective

- Define a utility function $U(X_s)$
  - increasing, concave
  - e.g. $U(X_s) = \log X_s$
  - no flow's rate can be too large or too small (fair)
  - logarithmic utility corresponds to "proportional fairness"



- Let us consider the problem of

$$\max \sum_s U(X_s)$$

$$\text{sub to } \left[\sum_s H_s X_s\right] \in \Lambda \triangleq \sum_k \lambda_k \, \text{Conv-hull}\{\vec{g}(\vec{P}, k) \mid \vec{P} \in \mathcal{P}\}$$

- Let $[X_s^*]$ be this optimal vector.

Drift + Penalty - 15min

Sunday, February 16, 2020 3:42 PM

Let us begin with the same Lyapunov function as before:

$$V(\vec{q}) = \frac{1}{2} \sum_l (q^l)^2$$

Since

$$\Rightarrow \quad (q^l(t+1))^2 \le \left(q^l(t) + \sum_s H_s^l x_s - r^l(t)\right)^2$$

$$= (q^l(t))^2 + 2q^l(t)\left[\sum_s H_s^l x_s - r^l(t)\right]$$

$$+ \left[\sum_s H_s^l x_s - r^l(t)\right]^2$$

Since x_s is finite, and $r^l(t)$ is bounded, there exists a constant M_l such that

$$(q^l(t+1))^2 \le (q^l(t))^2 + 2q^l(t)\left[\sum_s H_s^l x_s + r^l(t)\right] + M_l$$

$$\Rightarrow \quad V(\vec{q}(t+1)) \le V(\vec{q}(t)) + \sum_l q^l(t)\left[\sum_s H_s^l x_s - r^l(t)\right] + \frac{1}{2}\sum M_l^2.$$

$$\Rightarrow \quad E\left[V(\vec{q}(t+1)) - V(\vec{q}(t)) \,\middle|\, \vec{q}(t)\right]$$

$$\le \sum_l q^l(t) \sum_s H_s^l x_s - \sum_l q^l(t) E\left[r^l(t) \,|\, q^l(t)\right] + \frac{1}{2}\sum M_l^2.$$

However, maximizing $\sum_l q^l(t) E\left[r^l(t) / q^l(t)\right]$ does not help us to choose x_s.

Drift + Penalty

— Add the penalty term

$$\Delta(t) = \sum_s U(x_s^*) - \sum_s U(x_s)$$

- How suboptimal $[x_s]$ is.

- The sum is

$$V(\vec{q}(t+1)) - V(\vec{q}(t)) + \alpha\Delta(t)$$

$$\le \underbrace{\left[\sum_l q^l(t) \sum_s H_s^l x_s - \alpha \sum_s U(x_s)\right]}_{A}$$

$$- \underbrace{\sum_l q^l(t)\, E[r^l(t) \mid q^l(t)]}_{B} + \frac{1}{2}\sum M_l^2 + \alpha \sum_{i,c} U(\lambda_i^{c,*})$$

- We again try to minimize the drift-plus-penalty

- Minimizing A is the same as

$$\max_{[x_s]} \alpha \sum_s U(x_s) - \sum_l q^l(t) \sum_s H_s^l x_s$$

$$= \max_{[x_s]} \sum_s \left[\alpha U(x_s) - x_s \cdot \sum_l H_s^l q^l(t)\right]$$

$$= \sum_s \max_{x_s} \left[\alpha U(x_s) - x_s \cdot \sum_l H_s^l q^l(t)\right]$$

- Hence, each flow chooses x_s

$$x_s(t) = \arg\max_{x_s} \alpha U(x_s) - x_s \cdot \sum_l H_s^l q^l(t)$$

- We may interpret $\frac{q^l(t)}{\alpha}$ as a "price" of link l

- $\sum_l H_s^l \frac{q^l(t)}{\alpha}$ is the total "price" along the path of flow s.

- $x_s(t)$ is thus chosen to maximize the net utility.

- For $U(x_s) = \log x_s$

$$\Rightarrow \frac{\alpha}{x_s} = \sum_s H_s^l q^l(t) \Rightarrow \quad x_s = \frac{\alpha}{\sum_s H_s^l q^l(t)}$$

- maximizing B leads to the same max-weight scheduling decision.

Why does this work?

- Suppose that we choose $x_s(t) = x_s^*$.
- Then, since $[x_s^*] \in \Lambda$, we must have

$$E\left[V(\vec{q}(t+1)) - V(\vec{q}(t)) \middle| \vec{q}(t) \right]$$
$$\leq \sum_l q^l(t) \sum_s H_s^l x_s - \sum_l q^l(t) E\left[r^l(t) / q^l(t) \right] + \frac{1}{2} \sum M_l^2 .$$
$$\leq \frac{1}{2} \sum M_l^2$$

- Further,

$$\Delta(t) = \sum_s U(x_s^*) - \sum_s U(x_s)$$
$$= 0$$

- Together, the total drift + penalty would have been

$$V(\vec{q}(t+1)) - V(\vec{q}(t)) + \alpha \Delta(t)$$
$$\leq \frac{1}{2} \sum M_l^2$$

- Now, since our choice of $x_s(t)$ & $r^l(t)$ minimize the drift + penalty, the same must be true for our decision (even though we do not know x_s^*).

- Summing over all t, divided by T

$$\frac{E\left[V(\vec{q}(T+1))\right]-E\left[V(\vec{q}(0))\right]}{T}$$

$$+\alpha\sum_s U(x_s^*)-\alpha\frac{1}{T}\sum_{t=1}^{T}\sum_s E\left[U(x_s(t))\right]$$

$$\leq M \triangleq \frac{1}{2}\sum_l M_l^2$$

$$\Rightarrow \frac{1}{T}\sum_{t=1}^{T}\sum_s E\left(U(x_s(t))\right)$$

$$\geq \sum_s U(x_s^*) - \underbrace{\frac{M}{\alpha}}_{\to 0 \text{ as } \alpha\to+\infty} + \underbrace{\frac{E\left[V(\vec{q}(T+1))\right]}{\alpha T}}_{\geq 0} - \underbrace{\frac{E\left[V(\vec{q}(0))\right]}{\alpha T}}_{\to 0 \text{ as } T\to+\infty}$$

- As $\alpha\to+\infty$, the loss of average utility $\to 0$.

Reference:

Michael J. Neely , Eytan Modiano , Chih-ping Li, "Fairness and Optimal Stochastic Control for Heterogeneous Networks," *IEEE/ACM Transactions on Networking*, April 2008

Drift + Penalty - handout

Sunday, February 16, 2020 3:42 PM

Let us begin with the same Lyapunov function as before:

$$V(\vec{q}) = \frac{1}{2}\sum_l (q^l)^2$$

Since

$$q^l(t+1) = \left[q^l(t) + \sum_s H_s^l x_s - r^l(t)\right]^+$$

$$\Rightarrow \left(q^l(t+1)\right)^2 \le \left(q^l(t) + \sum_s H_s^l x_s - r^l(t)\right)^2$$

$$= \left(q^l(t)\right)^2 + 2q^l(t)\left[\sum_s H_s^l x_s - r^l(t)\right] + \left[\sum_s H_s^l x_s - r^l(t)\right]^2$$

Since x_s is finite, bounded, there exists a constant M_l such that

$$\left(q^l(t+1)\right)^2 \le \left(q^l(t)\right)^2 + 2q^l(t)\left[\sum_s H_s^l x_s + r^l(t)\right] + M_l$$

$$\Rightarrow V(\vec{q}(t+1)) \le V(\vec{q}(t)) + \sum_l q^l(t)\left[\sum_s H_s^l x_s - r^l(t)\right] + \frac{1}{2}\sum M_l^2.$$

$$\Rightarrow E\left[V(\vec{q}(t+1)) - V(\vec{q}(t)) \middle| \vec{q}(t)\right]$$

$$\le \sum_l q^l(t)\sum_s H_s^l x_s - \sum_l q^l(t)E\left[r^l(t)\middle| q^l(t)\right] + \frac{1}{2}\sum M_l^2.$$

However, maximizing $\sum_l q^l(t)E[r^l(t)/q^l(t)]$ does not help us to choose x_s.

Drift + Penalty

- Add the penalty term

$$\Delta(t) = \sum_s U(x_s^*) - \sum_s U(x_s)$$

- How suboptimal $[x_s]$ is.

- The sum is

$$V(\vec{q}(t+1)) - V(\vec{q}(t)) + \alpha \Delta(t)$$

$$\leq$$

- We again try to minimize the drift-plus-penalty

- Minimizing A is the same as

$$\max_{[x_s]} \alpha \sum_s U(x_s) - \sum_l q^l(t) \sum_s H_s^l x_s$$

$$=$$

$$=$$

- Hence, each flow chooses x_s

$$x_s(t) = \arg\max_{x_s} \alpha U(x_s) - x_s \cdot \sum_l H_s^l q^l(t)$$

- For $U(x_s) = \log x_s$

$$\Rightarrow \frac{\alpha}{x_s} = \sum_s H_s^l q^l(t) \Rightarrow \quad x_s = \frac{\alpha}{\sum_s H_s^l q^l(t)}$$

- maximizing B leads to the same max-weight scheduling decision.

Why does this work?

- Suppose that we choose $X_s(t) = X_s^*$.
- Then, since $[X_s^*] \in \Lambda$, we must have

$$E\left[V(\vec{q}(t+1)) - V(\vec{q}(t)) \,\middle|\, \vec{q}(t) \right]$$

- Further,

$$\Delta(t)$$

- Together, the total drift + penalty would have been

$$V(\vec{q}(t+1)) - V(\vec{q}(t)) + \alpha \Delta(t)$$
$$\leq \frac{1}{2} \sum M_i^2$$

- Now, since our choice of $X_s(t)$ & $r^l(t)$ minimize the drift + penalty, the same must be true for our decision (even though we do not know X_s^*).

— Summing over all t, divided by T

$$\Rightarrow \frac{1}{T}\sum_{t=1}^{T}\sum_s E\left[U(x_s(t))\right]$$

$$\geq \sum_s U(x_s^*) - \frac{M}{\alpha} - \underbrace{\frac{E\left[V(\vec{q}(0))\right]}{\alpha T}}_{\to 0 \text{ as } T\to+\infty} + \underbrace{\frac{E\left[V(\vec{q}(T+1))\right]}{\alpha T}}$$

— As $\alpha \to +\infty$, the loss of average utility $\to 0$.

Queue vs. Optimality tradeoff - 5min

Sunday, February 16, 2020 4:03 PM

<u>Queue</u>/Optimality tradeoff

- As $\alpha \to +\infty$, we expect that $x_s(t)$ is around x_s^*

- From (A)

$$\max \alpha U(x_s) - x_s \sum_l H_s^l q^l(t)$$

$$\Rightarrow \alpha U'(x_s) = \sum_l H_s^l q_l(t)$$

- As $\alpha \uparrow$, so is $q^l(t) \uparrow$.
  Closer optimality is achieved at the cost of higher queue length.

- Similar approach can be applied to other objectives, such as energy consumption.

Back-pressure - 15min

Friday, February 01, 2008 3:40 PM

- Optimal Joint Routing & Scheduling

Model:

- Link model same as that in the scheduling problem

$$\vec{r} = g(\vec{P}, K(t))$$

$$\vec{P} \in \Theta$$

$K(t)$ i.i.d over time.

- Multiple commodities $c = 1, 2, \cdots C$, each with a destination node $d(c)$

- A number of nodes may generate packets of commodity c.

λ_i^c = rate of new packets of commodity c generated by node i

- No routes are specified yet.

- In order for the arrival rate vector $[\lambda_i^c]$ to be supported, there must exist $[\bar{r}_{ij}^c]$ such that

- $\bar{r}_{ij}^c$ is the rate that node i forwards commodity c to node j

- $$\lambda_i^c + \sum_{j \neq i} \bar{r}_{ji} \leq \sum_{j \neq i} \bar{r}_{ij}$$

for all node $i \neq d(c)$

- $$\left[\sum_c \bar{r}_{ij}^c\right] \in \Lambda \triangleq \sum_k \pi_k \, \text{Conv-Hull} \{ g(\vec{P}, k) \mid \vec{P} \in \Theta \}$$

Queue-length Based Policy

- Each node maintains multiple queues, one for each destination c
  q_i^c

  Let r_{ij}^c be the amount of packets of commodity c that are forwarded from node i to node j.

$$q_i^c(t+1) = \begin{cases} \left[q_i^c(t) + \sum_{j \neq i} r_{ji}^c + \lambda_i^c - \sum_{j \neq i} r_{ij}^c \right]^+ & \text{if } i \neq d(c) \\ = 0 & \text{if } i = d(c). \end{cases}$$

- We now see why working with the Lyapunov function also produces a joint routing & scheduling policy that is throughput-optimal

- Use the Lyapunov function

$$V(\vec{q}) = \frac{1}{2} \sum_{i,c} (q_i^c)^2$$

Note

$$\left(q_i^c(t+1)\right)^2 - \left(q_i^c(t)\right)^2$$

$$\leq 2 q_i^c(t) \left[\sum_{j \neq i} r_{ji}^c + \lambda_i^c - \sum_{j \neq i} r_{ij}^c \right] + \text{constant}$$

$$\Rightarrow \quad V(\vec{q}(t+1)) - V(\vec{q}(t))$$

$$\leq \sum_{i,c} q_i^c(t) \left[\sum_{j \neq i} r_{ji}^c + \lambda_i^c - \sum_{j \neq i} r_{ij}^c \right] + \text{constant}$$

$$= \sum_{i,c} q_i^c(t) \cdot \lambda_i^c - \underbrace{\sum_{(i,j) \in L, c} r_{ij}^c \left[q_i^c(t) - q_j^c(t) \right]}_{\text{should maximize this to minimize the drift.}}$$

- Give $\vec{r} = g(\vec{p}, K(t))$, we have $\sum_c r_{ij}^c = r_{ij}$

$\Rightarrow$ maximizing the last term implies that we should only use r_{ij}^c with the largest value of $(q_i^c(t) - q_j^c(t))$

- The last-term becomes $\sum_{ij} r_{ij} \max_c [q_i^c(t) - q_j^c(t)]$

$\Rightarrow$ should then choose $\vec{p}(t)$ to maximize this weighted sum.

Joint Routing & Scheduling Algo.

① Maximum differential backlog.

For each link (i,j), select the commodity c such that the value

$$q_i^c(t) - q_j^c(t)$$

is the largest.

Let $c_{ij}^*(t) = \arg\max_c q_i^c(t) - q_j^c(t)$

Let $W_{ij}(t) = q_i^{c_{ij}^*(t)} - q_j^{c_{ij}^*(t)}$ denote the maximum differential backlog.

② Scheduling.

Choose $\vec{p}(t)$ such that

$$\vec{p}(t) = \arg\max_{\substack{\vec{p} \in \Pi \\ \vec{r} = g(\vec{p}, K(t))}} \sum_{(i,j)} W_{ij}(t) \cdot r_{ij}$$

Let $\vec{r}(t) = g(\vec{p}(t), K(t))$

③ Routing:

On each link (i,j), route the commodity $c_{ij}^*(t)$ using the rate r_{ij} i.e.

$$r_{ij}^c(t) = \begin{cases} r_{ij}(t) & \text{if } c = c_{ij}^*(t) \\ 0 & \text{, otherwise} \end{cases}$$

Intuition: As packets are queued, the queue difference forms a "gradient", which points to the optimal direction to forward packets

Can be shown to achieve the largest set of offered loads $[\lambda_i^c]$.

Reference: Neely & Modiano.

Dynamic Power Allocation and Routing for Time Varying Wireless Networks, by M. J. Neely, E. Modiano and C. E. Rohrs, in IEEE INFOCOM, April 2003.

35