# Throughput-Competitive Online Routing -Awerbuch, Azar, Plotkin; FOCS 1993

**Abhishek Sinha**
Laboratory for Information and Decision Systems

MIT

Talk at: CNRG Meeting

March 10, 2016

# Outline

# Outline

## Set-up

- Given a wired network $G(V, E)$ $|V| = n$, capacity of edge $e \in E$ is $u(e)$.
- Connection requests come sequentially in an online fashion (no apriori probability distributions of arrivals).
- Each request demands certain amount of network resources (e.g., a source-to-destination connection with certain bandwidth for certain time-span.) and is willing to pay certain price, if serviced.
- We may either accept the request or reject it.
- *No Queuing*: Acceptance means guaranteed service.
- **Problem:** Make optimal admission decision and routing decisions.

# Set-up, formally

- The $i^{\text{th}}$ connection request is formally represented by the following tuple

$$\beta_i = (s_i, t_i, r_i(\tau), T^s(i), T^f(i), \rho(i))$$

  $s_i$: origin of connection.
  $t_i$: destination of connection.
  $T^s(i)$: starting time of service.
  $T^f(i)$: completion time of service.
  $r_i(\tau)$: traffic rate demanded between time $[T^s(i), T^f(i)]$. Assumed to be zero outside this interval.
  $\rho(i)$: utility received by the controller upon serving the request.

- Decision: The controller either accepts or rejects $\beta_i$. If accepted, it assigns a $s_i - t_i$ path $P_i$ to $\beta_i$, otherwise $P_i \leftarrow \phi$.

## Set-Up Contd.

- Capacity Constraints must be respected at all times. Define the load on edge $e$ before reception of $k^{\text{th}}$ request as

$$\lambda_e(\tau, k) = \sum_{e \in P_i, i<k} \frac{r_i(\tau)}{u(e)}$$

We require $\lambda_e(\tau, k) \leq 1, \forall e, k, \tau$.

- Regularity Assumption (1): Since we are looking for throughput-optimization, utility is approximately proportional to bandwidth-time product.
  In other words, define the duration of the $j^{\text{th}}$ connection-request $T(j) := T^f(j) - T^s(j)$. Then, there exists a universal constant $F$ such that

$$1 \leq \frac{1}{n} \frac{\rho(j)}{r_j(\tau) T(j)} \leq F$$

# Regularity Assumption (2) : small-sized requests

- Regularity Assumption (2): Define $T = \max_j T(j)$ and $\mu \stackrel{\text{def}}{=} 2nTF + 1$.
  We assume that individual requests for bandwidths is a small fraction of the capacity of the edges, i.e.

$$r_j(\tau) \leq \frac{\min_e(u(e))}{\log \mu}, \quad \forall j, \tau \in [T^s(j), T^f(j)]$$

  This assumption, in essence implies that requests are *fluid-like* and we can apply control in a fine-grained fashion.

# Regularity Assumption (2) : small-sized requests

- Regularity Assumption (2): Define $T = \max_j T(j)$ and $\mu \stackrel{\text{def}}{=} 2nTF + 1$.
  We assume that individual requests for bandwidths is a small fraction of the capacity of the edges, i.e.

$$r_j(\tau) \leq \frac{\min_e(u(e))}{\log \mu}, \quad \forall j, \tau \in [T^s(j), T^f(j)]$$

  This assumption, in essence implies that requests are *fluid-like* and we can apply control in a fine-grained fashion.

## Algorithm Overview

- As the $j^{\text{th}}$ connection-request comes, a weight-vector $w_e(j, \tau)$ is computed for all $\tau \in [T^s(j), T^f(j)]$.
- A shortest path is computed on the graph based on these weight-functions, summed over from $[T^s(j), T^f(j)]$, with cost $v(j)$.
- If the benefit for serving the request is more than the cost, i.e. $v(j) \leq \rho(j)$ then the request is served along the shortest computed path. Else, the request is rejected.

# Outline

# Admission Control and Routing

## Online Control Algorithm $\mathcal{A}$

- On the arrival of $j^{\text{th}}$ connection-request, associate a weight $c_e(\tau, j)$ for each edge $e$, which is exponential in the current-load $\lambda_e(\tau, j)$ (before the request has come)

$$c_e(\tau, j) = u(e)(\mu^{\lambda_e(\tau,j)} - 1), \quad \forall \tau \in [T^s(j), T^f(j)]$$

- Find a shortest $s(j) - t(j)$ path with the weight of the edge $e$ being $w_e = \sum_\tau \frac{r(\tau)}{u(e)} c_e(\tau, j)$.
- If the cost of the shortest-path is less than or equal to $\rho(j)$ then accept the request and route it along the computed shortest path, else reject it.

# Admission Control and Routing

## Online Control Algorithm $\mathcal{A}$

- On the arrival of $j^{th}$ connection-request, associate a weight $c_e(\tau, j)$ for each edge $e$, which is exponential in the current-load $\lambda_e(\tau, j)$ (before the request has come)

$$c_e(\tau, j) = u(e)(\mu^{\lambda_e(\tau, j)} - 1), \quad \forall \tau \in [T^s(j), T^f(j)]$$

- Find a shortest $s(j) - t(j)$ path with the weight of the edge $e$ being $w_e = \sum_\tau \frac{r(\tau)}{u(e)} c_e(\tau, j)$.
- If the cost of the shortest-path is less than or equal to $\rho(j)$ then accept the request and route it along the computed shortest path, else reject it.

## Properties of the Algorithm $\mathcal{A}$

- The algorithm is *online*, does not require any statistical information, and have low-complexity $\mathcal{O}(n^2 T)$.
- Guaranteed service on acceptance, no-queuing, online routing.
- Is competitively optimal (within $\mathcal{O}(\log n)$ factor) and is optimal among all online policies.

# Outline

# Analysis-I: Feasibility

First we need to show that the algorithm is *feasible*, i.e., it *always* respects the edge-capacity constraint.

# Analysis-I: Feasibility

First we need to show that the algorithm is *feasible*, i.e., it *always* respects the edge-capacity constraint.
Intuitively, it follows from the fact that the algorithm rejects any request whose routing cost exceeds the benefit and that any request is of small size.

**Lemma (Feasibility of the Online Algorithm)**

*For all edges $e \in E$ and at all times $\tau$, we have*

$$\sum_{i \in \mathcal{A}, e \in P_i} r_i(\tau) \leq u_e \tag{1}$$

# Proof of Feasibility

Assume that $\beta_j$ be the first connection that was accepted and caused the relative load of edge $e$ to exceed 1.

Hence by definition, there is a slot $\tau \in [T^s(j), T^f(j)]$ such that $\lambda_e(\tau, j) > 1 - \frac{r_j(\tau)}{u(e)}$ (so that the edge overload). Let us estimate the cost at which the edge $e$ was included in the path

$$\begin{aligned} c_e(\tau, j)/u(e) &= \mu^{\lambda_e(\tau,j)} - 1 \\ &\geq \mu^{1-\frac{r_j(\tau)}{u(e)}} - 1 \\ &\overset{(a)}{\geq} \mu^{1-\frac{1}{\log(\mu)}} - 1 \\ &= \frac{\mu}{2} - 1 = TFn \end{aligned}$$

Where $(a)$ follows from the small rate assumption of individual requests.

Hence, the cost of the edge at time $\tau$ alone is $= \frac{c_e(\tau,j)}{u(e)} r_j(\tau) = r_j(\tau) TFn \overset{(b)}{\geq} \rho(j)$, where $(b)$ follows from the bounds on benefits. Thus, the $j^{th}$ connection-request violates the criteria for admission and concludes the proof of feasibility.

# Competitive Ratio

**Theorem**

*The online algorithm $\mathcal{A}$ is optimal within a multiplicative-factor of $\mathcal{O}(\log n)$.*

This theorem is proved in two simple lemmas.

**Lemma ( Lower-bound on Accumulated profit)**

*Let $\mathcal{I}$ be the set of indices of connection accepted by the online algorithm and let $k$ be the index of the last connection, then*

$$\sum_{j \in \mathcal{I}} \rho(j) \geq \frac{1}{2 \log \mu} \sum_{\tau} \sum_{e} c_e(\tau, k+1) \quad (2)$$

# Competitive Ratio

**Theorem**

*The online algorithm $\mathcal{A}$ is optimal within a multiplicative-factor of $\mathcal{O}(\log n)$.*

This theorem is proved in two simple lemmas.

**Lemma ( Lower-bound on Accumulated profit)**

*Let $\mathcal{I}$ be the set of indices of connection accepted by the online algorithm and let $k$ be the index of the last connection, then*

$$\sum_{j \in \mathcal{I}} \rho(j) \geq \frac{1}{2 \log \mu} \sum_{\tau} \sum_{e} c_e(\tau, k+1) \qquad (2)$$

Finally, it is shown that the profit of the requests left out by $\mathcal{A}$ but accepted by the off-line optimal algorithm can not be large.

# Competitive Ratio

**Theorem**

*The online algorithm $\mathcal{A}$ is optimal within a multiplicative-factor of $\mathcal{O}(\log n)$.*

This theorem is proved in two simple lemmas.

**Lemma ( Lower-bound on Accumulated profit)**

*Let $\mathcal{I}$ be the set of indices of connection accepted by the online algorithm and let $k$ be the index of the last connection, then*

$$\sum_{j \in \mathcal{I}} \rho(j) \geq \frac{1}{2 \log \mu} \sum_{\tau} \sum_{e} c_e(\tau, k+1) \tag{2}$$

Finally, it is shown that the profit of the requests left out by $\mathcal{A}$ but accepted by the off-line optimal algorithm can not be large.

**Lemma (Upper-bound on Relative loss)**

*Let $\mathcal{Q}$ be the set of indices of the connections that were admitted by the off-line algorithm but were rejected by the on-line algorithm. Denote $l = \max\{\mathcal{Q}\}$. Then*

$$\sum_{j \in \mathcal{Q}} \rho(j) \leq \sum_{\tau} \sum_{e} c_e(\tau, l) \tag{3}$$

# Proof of Lemma 1: Lower-bound on the accumulated profit

Suppose that we admit the $j^{\text{th}}$ connection request $\beta_j$. Since the requests are small, cost of an edge should not change much because of its admission. In particular, consider an edge $e \in \mathcal{P}_j$. The change in cost can be calculated as follows :

# Proof of Lemma 1: Lower-bound on the accumulated profit

Suppose that we admit the $j^{th}$ connection request $\beta_j$. Since the requests are small, cost of an edge should not change much because of its admission. In particular, consider an edge $e \in \mathcal{P}_j$. The change in cost can be calculated as follows :

$$\begin{aligned} c_e(\tau, j+1) - c_e(\tau, j) &= u(e)(\mu^{\lambda_e(\tau,j)+\frac{r_j(\tau)}{u(e)}} - \mu^{\lambda_e(\tau,j)}) \\ &= u(e)\mu^{\lambda_e(\tau,j)}(2^{\log(\mu)\frac{r_j(\tau)}{u(e)}} - 1) \end{aligned}$$

By our assumption of small requests (i.e., $\frac{r_j(\tau)}{u(e)} \leq \frac{1}{\log(\mu)}$), and the fact that $2^x - 1 \leq x$ for $0 \leq x \leq 1$, we conclude that

$$c_e(\tau, j+1) - c_e(\tau, j) \leq c_e(\tau, j)\frac{r_j(\tau)}{u(e)} \log \mu$$

Summing over all $e$ and $\tau$ and using the fact that $\beta_j$ was admitted, we have

$$\sum_{e,\tau}[c_e(\tau, j+1) - c_e(\tau, j)] \leq \log \mu \sum_{e \in \mathcal{P}_j, \tau} c_e(\tau, j)\frac{r_j(\tau)}{u(e)} \leq \rho(j) \log \mu$$

Summing over all $j \in \mathcal{I}$ completes the proof. ■

# Proof of Lemma 2: Upper-bound on Relative Loss

Since load at an edge at a slot can only increase with more requests, we have for all $c_e(\tau, j) \leq c_e(\tau, l), \forall j, e, \tau$. Consider a request $j \in \mathcal{Q}$. Since it was rejected by the online algorithm, we must have

$$\rho(j) \leq \sum_{\tau} \sum_{e \in P'_j} r_j(\tau) c_e(\tau, j)/u(e) \leq \sum_{\tau} \sum_{e \in P'_j} r_j(\tau) c_e(\tau, l)/u(e)$$

Summing over all $j \in \mathcal{Q}$, we have

$$\sum_{j \in \mathcal{Q}} \rho(j) \leq \sum_{\tau} \sum_{e} c_e(\tau, l) \sum_{j: e \in P'_j} \frac{r_j(\tau)}{u(e)} \overset{(*)}{\leq} \sum_{\tau, e} c_e(\tau, l)$$

where $(*)$ follows because the offline algorithm is not allowed to over load the edge at any slot. ■

# Proof of Lemma 2: Upper-bound on Relative Loss

Since load at an edge at a slot can only increase with more requests, we have for all $c_e(\tau, j) \leq c_e(\tau, l), \forall j, e, \tau$. Consider a request $j \in \mathcal{Q}$. Since it was rejected by the online algorithm, we must have

$$\rho(j) \leq \sum_{\tau} \sum_{e \in P'_j} r_j(\tau) c_e(\tau, j)/u(e) \leq \sum_{\tau} \sum_{e \in P'_j} r_j(\tau) c_e(\tau, l)/u(e)$$

Summing over all $j \in \mathcal{Q}$, we have

$$\sum_{j \in \mathcal{Q}} \rho(j) \leq \sum_{\tau} \sum_{e} c_e(\tau, l) \sum_{j : e \in P'_j} \frac{r_j(\tau)}{u(e)} \overset{(*)}{\leq} \sum_{\tau, e} c_e(\tau, l)$$

where $(*)$ follows because the offline algorithm is not allowed to over load the edge at any slot. ■

**Proof of Approximation Guarantee:** Combine the above two lemmas. ■