Online
Path Computation & Function Placement in SDNs

Guy Even
Tel Aviv University

Moti Medina
MPI for Informatics

Boaz Patt-Shamir
Tel Aviv University
Today’s Focus: Online Virtual Circuit Routing

• Network: \( G = (V, E, c) \)
  - \( V \) is the set of nodes
  - \( E \) is the set of links
  - \( c: E \to \mathbb{N}^+ \) edge capacities

• Requests: paths \( r_j = \langle s_j, t_j, b_j, d_j, [\alpha_j, \beta_j] \rangle \)
  - \( s_j \) source node
  - \( t_j \) destination node
  - \( b_j \) benefit (per time step)
  - \( d_j \) BW requirement
  - \( [\alpha_j, \beta_j] \) arrival/depart times
    - \( \beta_j \) is either known or unknown upfront.

\[
\text{load}(e) \triangleq \frac{\text{flow}(e)}{c(e)}
\]
Performance Measure - Competitive Ratio
[Sleator, Tarajan 85]

• $ALG$ : Online alg
• $\sigma$ : sequence of path requests
• $ALG(\sigma)$ : total benefit due to $\sigma$
• $OPT(\sigma)$ : Max benefit from $\sigma$ by a feasible allocation

$$\rho(ALG) \triangleq \inf_{\sigma} \frac{ALG(\sigma)}{OPT(\sigma)}$$
## Online Versions

<table>
<thead>
<tr>
<th>Duration</th>
<th>Model</th>
<th>Benefits:</th>
</tr>
</thead>
<tbody>
<tr>
<td>known</td>
<td>ACC/REJECT</td>
<td>$C.R. = \log(</td>
</tr>
<tr>
<td></td>
<td>ACC/STDBY</td>
<td>Same as ACC/REJECT</td>
</tr>
<tr>
<td>unknown</td>
<td>ACC/REJECT</td>
<td>$C.R. \geq \text{time}$</td>
</tr>
<tr>
<td></td>
<td>ACC/STDBY</td>
<td>New: Same as ACC/REJECT</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Benefits:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{ACC/REJECT} = \sum_{\text{accepted}} b_j$</td>
</tr>
<tr>
<td>$\text{ACC/STDBY} = \sum_{\text{accepted}(t)} b_j$</td>
</tr>
<tr>
<td>Known duration $\approx$ only arrivals (persistent requests)</td>
</tr>
</tbody>
</table>
Proposal: ACC/STDBY model

- Unknown duration
- No preemption (!)
- No rerouting (!)
- Request pays $b_j$ per served time unit.

- Question: what about online competitive analysis?
# Online Versions

<table>
<thead>
<tr>
<th>Model</th>
<th>Duration</th>
<th>Benefits:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACC/REJECT</td>
<td>known</td>
<td>$C.R. = \log(</td>
</tr>
<tr>
<td></td>
<td>[AAP93]</td>
<td>$ACC/REJECT = \sum_{accepted} b_j$</td>
</tr>
<tr>
<td></td>
<td>unknown</td>
<td>$ACC/STDBY = \sum_t \sum_{accepted(t)} b_j$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Known duration $\approx$ only arrivals (persistent requests)</td>
</tr>
<tr>
<td>ACC/STDBY</td>
<td></td>
<td>New: $C.R. = \log(</td>
</tr>
</tbody>
</table>
Online Persistent Routing Algorithm [AAP93]
(known duration, ACC/REJ)

- **State**: accepted requests \( \{r_j, path_j\}_{acc} \)
  
- **\( r_j \) arrives**: \( r_j \) arrives

  ![Diagram]

  - **Path Oracle**
  - **Allocate** \( path_j \) to \( r_j \)

  - **Oracle** \( (r_j) \)
    - \( w(e) = \exp(\text{load}(e)) \)
    - Search for a path \( s_j \overset{\curvearrowright}{\rightarrow} t_j \)
      s.t. \( \sum_{e \in path_j} w(e) < b_j \)
    - Success: return \( (path_j) \)
    - Fail: REJECT

  \[
  \text{load}(e) \equiv \frac{\text{flow}(e)}{\text{c}(e)}
  \]
Our Algorithm
(unknown duration, ACC/STDBY)

- **State**: accepted requests \( \{r_j, path_j\}_{acc.} \), STDBY list.
- \( r_j \) arrives:
  - **Oracle** \( r_j \):
    - \( w(e) = \exp(\text{load}(e)) \)
    - Search for a path \( s_j \rightarrow t_j \) s.t. \( \sum_{e \in \text{path}_j} w(e) < b_j \)
    - Success: return(\( path_j \))
    - Fail: **STDBY**
  - \( r_j \) arrives: ACCEPT
  - Allocate \( path_j \) to \( r_j \)

\[ \text{load}(e) \triangleq \frac{\text{flow}(e)}{c(e)} \]
Main Result (virtual circuit routing version)

**Thm.**
Small demands

\[ \implies \]

in each time step \( t \),

\[ \text{benefit}_t(\text{ALG}) \geq \frac{\text{benefit}_t(\text{OPT})}{\log(|V|b_{\text{max}})} \]

**Remarks:**
1. Small demands: \( d_{\text{max}} \leq \frac{c_{\text{min}}}{\log(|V|b_{\text{max}})} \).
2. If served, request is served continuously until it ends.
3. Even when \( \text{OPT} \) is fractional, may preempt, may reroute, i.e., MCF in every time step.
4. Proof idea: repeating the “game” of persistent routing.
5. Asymptotically optimal!
6. Extends to the SDN modeling (next slide).
Modeling SDN

• Network \( N = (V, E) \)
  • \( V \) set of servers
  • \( E \) set of links
  • \( c: V \cup E \rightarrow \mathbb{N}^+ \)
    • i.e., Processing, BW
• Online requests \( r_j = \{G_j, d_j, b_j, U_j\} \)
  • \( G_j = (X_j, Y_j) \) PR-graph
  • \( d_j: X_j \cup Y_j \rightarrow \mathbb{N} \) demand
  • \( b_j \in \mathbb{N}^+ \): benefit per served time step
  • \( U_j: X_j \cup Y_j \rightarrow 2^V \cup 2^E \) mapping

\[ e \text{ can be implemented by } \]
\[ U(x_1) = \{v_{17}, v_{23}, v_{97}, \ldots\} \]

Can be implemented by
\[ v_0 \xrightarrow[p]{e} v_k \text{ if } \forall i (v_{i-1}, v_i) \in U(e) \]

Action vertices – enc/dec, comp/decomp...

Single source – packets arrive from here

Single sink – packets are destined to here

Req served by any \( s - t \) path. Needs mapping to \( N \)
Examples

Simple Routing

“I want to route a connection of 100 mbps from $v$ to $v'$ “

• PR-graph $s \rightarrow t$
• $U(s) = \{v\}$,
• $U(t) = \{v'\}$,
• $U(e) = E$,
• $d(e) = 120 \text{ mbps}$

Serial Processing

• Stream,
• Pass $k$ transformations $a_1, \ldots, a_k$ in series.

• PR-graph $s \rightarrow a_1 \rightarrow \cdots \rightarrow a_k \rightarrow t$
• $U(a_i) \subseteq V$ implements $a_i$
• Can model BW changes
  • E.g., $a_j$ is COMPRESS.
  • How? Set diff demands to diff PR-edges.
Summary

• Need to allocate network resources in online fashion.
• Each request specifies sequence(s) of functions.
• Map desired functions to real network devices.
• Route the request between these devices.
• Security: want to avoid some nodes.
• Flexibility: customer does not want to commit to the duration upfront → the duration is unknown.
• Maximum benefit: maximizing the benefit rate.
That's all Folks!