OptFlow: A Flow-based Abstraction for Programmable Topologies

Klaus-Tycho Foerster  
Faculty of Computer Science  
University of Vienna  
Austria

Long Luo  
University of Electronic Science and Technology of China  
P.R. China

Manya Ghobadi  
Computer Science and Artificial Intelligence Laboratory, MIT  
USA

Wide/Regional/Metro-Area Networks are not Static
Wide/Regional/Metro-Area Networks are not Static
Closer look: Optical components

- Dense Wavelength Division Multiplexing (DWDM)
  - > 100 wavelengths (e.g. 100Gbps) per fiber
Closer look: Optical components

• Dense Wavelength Division Multiplexing (DWDM)
  - > 100 wavelengths (e.g. 100Gbps) per fiber

• Wavelengths can be steered at connection points
  ◦ By ROADMs (reconfigurable add/drop multiplexers)
Intuition: Move wavelengths
Intuition: Move wavelengths

\[ \text{wavelength capacity} = 1 \]
Intuition: Move wavelengths

![Diagram]

wavelength capacity = 1
Intuition: Move wavelengths

wavelength capacity = 1

throughput: 1+1
Intuition: Move wavelengths

![Diagram showing wavelength capacity and throughput]

- Wavelength capacity = 1
- Throughput: 1+1
- Throughput: 2+2
Intuition: Move wavelengths

Demand-aware capacity (wavelengths)
Intuition: Move wavelengths

Topology Programmability (TP) + Traffic Engineering (TE) > TE

demand-aware capacity (wavelengths)
How to leverage Topology Programmability (TP)?

- Variant #A: Optimize TP and Traffic Engineering (TE) separately?
  - Inefficient, misses opportunities (recall last slide)

- Variant #B: Redesign TE to include TP?
  - Tedious, operators are reluctant

- Variant #C: Don’t change TE, still incorporate TP!
  - Abstractions!
How to incorporate current Traffic Engineering (TE)?

Abstractions

Unmodified Traffic Eng. \[\downarrow\] Augmented Topology

Flow routing
Reconfiguration
How to design the Abstraction?

Setting in our example:
- Every node supports 2 wavelengths
- Every edge supports 2 wavelengths
How to design the Abstraction?

Setting in our example:
- Every node supports 2 wavelengths
- Every edge supports 2 wavelengths

Idea: u should only send 2 real units of traffic
- Implement fake flows that block capacity
  - Represent dual wavelength assignment
  - TEs can deal with flows
How to design the Abstraction?

Setting in our example:
• Every node supports 2 wavelengths
• Every edge supports 2 wavelengths

Idea: u should only send 2 real units of traffic
• Implement fake flows that block capacity
  • Represent dual wavelength assignment
  • TEs can deal with flows

TE performs TP by routing both flow types
• Fake flows from u to x or v
• Real traffic from u to x
Intuition for the Abstraction

Setting in our example:

- Every node supports 2 wavelengths
- Every edge supports 2 wavelengths

- \( u \) wants to send traffic to \( x \)
- \( v \) wants to send traffic to \( w \)
Intuition for the Abstraction

Setting in our example:
• Every node supports 2 wavelengths
• Every edge supports 2 wavelengths
• $u$ wants to send traffic to $x$
• $v$ wants to send traffic to $w$

Abstraction intuition
• Every node sends 2 unit flows to neighbors
• Every edge has a capacity of 2
• $u$ still wants to send traffic to $x$
• $v$ still wants to send traffic to $w$
Intuition for the Abstraction

Setting in our example:
• Every node supports 2 wavelengths
• Every edge supports 2 wavelengths
• u wants to send traffic to x
• v wants to send traffic to w

Abstraction intuition:
• Every node sends 2 unit flows to neighbors
• Every edge has a capacity of 2
• u still wants to send traffic to x
• v still wants to send traffic to w

Result:
• 2 fake flows between u,v & x,w
• No capacity left between u,v & x,w
• 2 units of capacity for u,v & x,w
• Real throughput of: 2+2
Intuition for the Abstraction

Setting in our example:
• Every node supports 2 wavelengths
• Every edge supports 2 wavelengths
• \( u \) wants to send traffic to \( x \)
• \( v \) wants to send traffic to \( w \)

Abstraction intuition
• Every node sends 2 unit flows to neighbors
• Every edge has a capacity of 2
• \( u \) still wants to send traffic to \( x \)
• \( v \) still wants to send traffic to \( w \)

Result:
• 2 fake flows between \( u,v \) & \( x,w \)
• No capacity left between \( u,v \) & \( x,w \)
• 2 units of capacity for \( u,v \) & \( x,w \)
• Real throughput of: 2+2
TE performs TE+TP on abstraction details in the paper

Intuition for the Abstraction

Setting in our example:
- Every node supports 2 wavelengths
- Every edge supports 2 wavelengths
- u wants to send traffic to x
- v wants to send traffic to w

Abstraction intuition
- Every node sends 2 unit flows to neighbors
- Every edge has a capacity of 2
- u still wants to send traffic to x
- v still wants to send traffic to w

Result:
- 2 fake flows between u,v & x,w
- No capacity left between u,v & x,w
- 2 units of capacity for u,v & x,w
- Real throughput of: 2+2

TE performs TE+TP on abstraction details in the paper
Takeaway

• TE performs TE and TP due to the abstraction
  ◦ Details in the paper

• Consistent update methods for flows carry over
  ◦ Abstraction enables cross-layer updates for free

• Support for major TE types (max. throughput, k-shortest path routing etc.)
Testbed: Demonstration of TP in Practice

Physical setup of our testbed

Logical setup
Experiment: Demonstration of TP in Practice

- Traffic from A to C (via A-C and A-B-D-C)
Experiment: Demonstration of TP in Practice

- Traffic from A to C (via A-C and A-B-D-C)
- Fail A-B link
**Experiment: Demonstration of TP in Practice**

- Traffic from A to C (via A-C and A-B-D-C)
- Fail A-B link
- Controller notices cut & shifts wavelength

![Graph showing throughput comparison with and without programmability.](image)

---

OptFlow: A Flow-based Abstraction for Programmable Topologies (SOSR’20)

---

13
Simulations: k-Shortest Path Routing
Simulations: k-Shortest Path Routing

- Comparison: Standard ILP (JointOpt) vs. our approach (OptFlow)

<table>
<thead>
<tr>
<th>Topology</th>
<th>#Nodes</th>
<th>#Links</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google (G-Scale) [28]</td>
<td>12</td>
<td>38</td>
</tr>
<tr>
<td>Internet2 [30]</td>
<td>40</td>
<td>100</td>
</tr>
<tr>
<td>IDN [26]</td>
<td>40</td>
<td>390</td>
</tr>
<tr>
<td>AS 1221 (Telstra) [2]</td>
<td>104</td>
<td>306</td>
</tr>
</tbody>
</table>
Simulations: k-Shortest Path Routing

- Comparison: Standard ILP (JointOpt) vs. our approach (OptFlow)

OptFlow: A Flow-based Abstraction for Programmable Topologies (SOSR’20)
Simulations: k-Shortest Path Routing

- Comparison: Standard ILP (JointOpt) vs. our approach (OptFlow)

<table>
<thead>
<tr>
<th>Topology</th>
<th>#Nodes</th>
<th>#Links</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google (G-Scale)</td>
<td>12</td>
<td>38</td>
</tr>
<tr>
<td>Internet2 [30]</td>
<td>40</td>
<td>100</td>
</tr>
<tr>
<td>IDN [26]</td>
<td>40</td>
<td>390</td>
</tr>
<tr>
<td>AS 1221 (Telstra)</td>
<td>104</td>
<td>306</td>
</tr>
</tbody>
</table>
Summary and Outlook

• Our abstraction enables Traffic Engineering to leverage Topology Programming
  ◦ Main idea: represent reconfigurability as flows
  ◦ Key items evaluated in a small testbed
  ◦ Simulations show good run time performance

• Outlook: Expand to
  ◦ further Traffic Engineering objectives
  ◦ include amplifiers and long-range wavelengths
OptFlow: A Flow-based Abstraction for Programmable Topologies

Klaus-Tycho Foerster  
Faculty of Computer Science  
University of Vienna  
Austria

Long Luo  
University of Electronic Science and Technology of China  
P.R. China

Manya Ghobadi  
Computer Science and Artificial Intelligence Laboratory, MIT  
USA