ProgrammabilityMedic: Predictable Path Programmability Recovery under Multiple Controller Failures in SD-WANs

Songshi Dou, Zehua Guo, and Yuanqing Xia

School of Automation
Beijing Institute of Technology

ICDCS’21 (July 8, 2021)
1. Background and Motivation
   1.1. Software-Defined Networking (SDN)
   1.2. Software-Defined Wide Area Networks (SD-WANs)
   1.3. Controller failure problem in SD-WANs

2. Existing Solutions and Limitations
   2.1. Switch-level mapping solution
   2.2. Flow-level mapping solution

3. Overview of ProgrammabilityMedic
   3.1. Opportunity
   3.2. Design considerations
   3.3. Flow Mode Selection and Switch Mapping problem
   3.4. Heuristic solution: ProgrammabilityMedic

4. Evaluation

5. Summary
1. Background and Motivation
   1.1. Software-Defined Networking (SDN)
   1.2. Software-Defined Wide Area Networks (SD-WANs)
   1.3. Controller failure problem in SD-WANs

2. Existing Solutions and Limitations
   2.1. Switch-level mapping solution
   2.2. Flow-level mapping solution

3. Overview of ProgrammabilityMedic
   3.1. Opportunity
   3.2. Design considerations
   3.3. Flow Mode Selection and Switch Mapping problem
   3.4. Heuristic solution: ProgrammabilityMedic

4. Evaluation

5. Summary
Advantages of SDN (e.g., path programmability).
Software-Defined Networking (SDN)

Advantages of SDN (e.g., path programmability).

Traditional networking

SDN controller

SDN switch

Software-Defined networking
Software-Defined Networking (SDN)

Fixed routing can not flexibly realize traffic scheduling

Traditional networking

SDN controller

SDN switch

Software-Defined networking

Advantages of SDN (e.g., path programmability).
Software-Defined Networking (SDN)

Fixed routing can not flexibly realize traffic scheduling

Traditional networking

SDN controller

SDN switch

Advantages of SDN (e.g., path programmability).
Software-Defined Networking (SDN)

Fixed routing cannot flexibly realize traffic scheduling.

Traditional networking

Flow table operation

SDN switch

Software-Defined networking

Advantages of SDN (e.g., path programmability).
Software-Defined Networking (SDN)

Fixed routing cannot flexibly realize traffic scheduling

Traditional networking

Controller can flexibly realize traffic scheduling

Software-Defined networking

Advantages of SDN (e.g., path programmability).
1. Background and Motivation
   1.1. Software-Defined Networking (SDN)
   1.2. Software-Defined Wide Area Networks (SD-WANs)
   1.3. Controller failure problem in SD-WANs

2. Existing Solutions and Limitations
   2.1. Switch-level mapping solution
   2.2. Flow-level mapping solution

3. Overview of ProgrammabilityMedic
   3.1. Opportunity
   3.2. Design considerations
   3.3. Flow Mode Selection and Switch Mapping problem
   3.4. Heuristic solution: ProgrammabilityMedic

4. Evaluation

5. Summary
Software-Defined Wide Area Networks (SD-WANs)

- Large scale with many devices
- Partitioning the network into domains
- Distributed control plane
  1. Quick response
  2. Control resiliency
  3. Controller synchronization

AT&T topology.
Software-Defined Wide Area Networks (SD-WANs)

- Large scale with many devices
- Partitioning the network into domains
- Distributed control plane
  1. Quick response
  2. Control resiliency
  3. Controller synchronization

AT&T topology.
Software-Defined Wide Area Networks (SD-WANs)

- Large scale with many devices
- Partitioning the network into domains
- Distributed control plane
  1. Quick response
  2. Control resiliency
  3. Controller synchronization

AT&T topology.
1. **Background and Motivation**
   1.1. Software-Defined Networking (SDN)
   1.2. Software-Defined Wide Area Networks (SD-WANs)
   1.3. **Controller failure problem in SD-WANs**

2. **Existing Solutions and Limitations**
   2.1. Switch-level mapping solution
   2.2. Flow-level mapping solution

3. **Overview of ProgrammabilityMedic**
   3.1. Opportunity
   3.2. Design considerations
   3.3. Flow Mode Selection and Switch Mapping problem
   3.4. Heuristic solution: ProgrammabilityMedic

4. **Evaluation**

5. **Summary**
Controller failure problem in SD-WANs

Controller failure
- Software bugs, Hardware failure, Power outage

Maintaining path programmability
- Offline switches remapping

Challenges
- Limited control resource
- Path programmability recovery
- Multiple failures

An example of controller failure.
Controller failure problem in SD-WANs

Controller failure
- Software bugs, Hardware failure, Power outage

Maintaining path programmability
- Offline switches remapping

Challenges
- Limited control resource
- Path programmability recovery
- Multiple failures

An example of controller failure.

D1
D2
D3
C1
C2
C3
s21
s20
s22
s24
s23
Failed controller

S. Dou et al.  ProgrammabilityMedic: Predictable Path Programmability Recovery under Multiple Controller Failures in SD-WANs  July 8, 2021 13 | 36
Controller failure problem in SD-WANs

**Controller failure**
- Software bugs, Hardware failure, Power outage

**Maintaining path programmability**
- Offline switches remapping

**Challenges**
- Limited control resource
- Path programmability recovery
- Multiple failures

An example of controller failure.
Outline

1. Background and Motivation
   1.1. Software-Defined Networking (SDN)
   1.2. Software-Defined Wide Area Networks (SD-WANs)
   1.3. Controller failure problem in SD-WANs

2. Existing Solutions and Limitations
   2.1. Switch-level mapping solution
   2.2. Flow-level mapping solution

3. Overview of ProgrammabilityMedic
   3.1. Opportunity
   3.2. Design considerations
   3.3. Flow Mode Selection and Switch Mapping problem
   3.4. Heuristic solution: ProgrammabilityMedic

4. Evaluation

5. Summary
Outline

1. Background and Motivation
   1.1. Software-Defined Networking (SDN)
   1.2. Software-Defined Wide Area Networks (SD-WANs)
   1.3. Controller failure problem in SD-WANs

2. Existing Solutions and Limitations
   2.1. Switch-level mapping solution
   2.2. Flow-level mapping solution

3. Overview of ProgrammabilityMedic
   3.1. Opportunity
   3.2. Design considerations
   3.3. Flow Mode Selection and Switch Mapping problem
   3.4. Heuristic solution: ProgrammabilityMedic

4. Evaluation

5. Summary
Switch-level mapping solution

RetroFlow*

- Some offline switches work under the **legacy routing mode** without the controllers
- The rest offline switches with the SDN routing mode

Limitations

- Fixed control cost of switches
- Per-switch mapping

---

Switch-level mapping solution

RetroFlow*

- Some offline switches work under the **legacy routing mode** without the controllers
- The rest offline switches with the SDN routing mode

Limitations

- Fixed control cost of switches
- Per-switch mapping

---


---

Switch with the SDN mode

Switch with the legacy mode

An example of switch-level solution.
1. Background and Motivation
   1.1. Software-Defined Networking (SDN)
   1.2. Software-Defined Wide Area Networks (SD-WANs)
   1.3. Controller failure problem in SD-WANs

2. Existing Solutions and Limitations
   2.1. Switch-level mapping solution
   2.2. Flow-level mapping solution

3. Overview of ProgrammabilityMedic
   3.1. Opportunity
   3.2. Design considerations
   3.3. Flow Mode Selection and Switch Mapping problem
   3.4. Heuristic solution: ProgrammabilityMedic

4. Evaluation

5. Summary
Flow-level mapping solution

Switch-level mapping solution

ProgrammabilityGuardian†

- Introducing a middle layer between controller and switches using Flowvisor
- Fine-grained per-flow mapping

Limitations

- Increasing the processing delay
- Bringing new unreliability

†Z. Guo et al., Improving the path programmability for software-defined wans under multiple controller failures, in IEEE/ACM IWQoS’20.
Flow-level mapping solution

Switch-level mapping solution

ProgrammabilityGuardian†

- Introducing a middle layer between controller and switches using Flowvisor
- Fine-grained per-flow mapping

Limitations

- Increasing the processing delay
- Bringing new unreliability

† Z. Guo et al., Improving the path programmability for software-defined wans under multiple controller failures, in IEEE/ACM IWQoS’20.
1. Background and Motivation
   1.1. Software-Defined Networking (SDN)
   1.2. Software-Defined Wide Area Networks (SD-WANs)
   1.3. Controller failure problem in SD-WANs

2. Existing Solutions and Limitations
   2.1. Switch-level mapping solution
   2.2. Flow-level mapping solution

3. Overview of ProgrammabilityMedic
   3.1. Opportunity
   3.2. Design considerations
   3.3. Flow Mode Selection and Switch Mapping problem
   3.4. Heuristic solution: ProgrammabilityMedic

4. Evaluation

5. Summary
1. Background and Motivation
   1.1. Software-Defined Networking (SDN)
   1.2. Software-Defined Wide Area Networks (SD-WANs)
   1.3. Controller failure problem in SD-WANs

2. Existing Solutions and Limitations
   2.1. Switch-level mapping solution
   2.2. Flow-level mapping solution

3. Overview of ProgrammabilityMedic
   3.1. Opportunity
   3.2. Design considerations
   3.3. Flow Mode Selection and Switch Mapping problem
   3.4. Heuristic solution: ProgrammabilityMedic

4. Evaluation

5. Summary
Opportunity

Hybrid SDN/legacy routing mode

- Specifically hybrid OpenFlow/OSPF routing mode
- Selectively deciding the routing mode for each offline flows
- Dynamically changing the control cost of offline flows
Outline

1. Background and Motivation
   1.1. Software-Defined Networking (SDN)
   1.2. Software-Defined Wide Area Networks (SD-WANs)
   1.3. Controller failure problem in SD-WANs

2. Existing Solutions and Limitations
   2.1. Switch-level mapping solution
   2.2. Flow-level mapping solution

3. Overview of ProgrammabilityMedic
   3.1. Opportunity
   3.2. Design considerations
   3.3. Flow Mode Selection and Switch Mapping problem
   3.4. Heuristic solution: ProgrammabilityMedic

4. Evaluation

5. Summary
Design considerations

Consideration 1:
• Recovering offline flows as many as possible

Consideration 2:
• Balancing path programmability

Consideration 3:
• Fully utilizing controllers’ control resource
Design considerations

Consideration 1:
• Recovering offline flows as many as possible

Consideration 2:
• Balancing path programmability

Consideration 3:
• Fully utilizing controllers’ control resource
Design considerations

Consideration 1:
• Recovering offline flows as many as possible

Consideration 2:
• Balancing path programmability

Consideration 3:
• Fully utilizing controllers’ control resource
Outline

1. Background and Motivation
   1.1. Software-Defined Networking (SDN)
   1.2. Software-Defined Wide Area Networks (SD-WANs)
   1.3. Controller failure problem in SD-WANs

2. Existing Solutions and Limitations
   2.1. Switch-level mapping solution
   2.2. Flow-level mapping solution

3. Overview of ProgrammabilityMedic
   3.1. Opportunity
   3.2. Design considerations
   3.3. Flow Mode Selection and Switch Mapping problem
   3.4. Heuristic solution: ProgrammabilityMedic

4. Evaluation

5. Summary
Flow Mode Selection and Switch Mapping problem

Switch-controller mapping constraint

• Each switch can be mapped to at most one controller

\[
\sum_{j=1}^{M} x_{ij} \leq 1, \forall \ i. \tag{1}
\]

Controller resource constraint

• The control load of a controller should not exceed the controller’s available control resource

\[
\sum_{l=1}^{L} \sum_{i=1}^{N} (x_{ij} \ast \beta_i^l \ast y_i^l) \leq A_j^{rest}, \forall \ j. \tag{2}
\]
Flow Mode Selection and Switch Mapping problem

Switch-controller mapping constraint

- Each switch can be mapped to at most one controller

\[
\sum_{j=1}^{M} x_{ij} \leq 1, \forall \ i. \tag{1}
\]

Controller resource constraint

- The control load of a controller should not exceed the controller’s available control resource

\[
\sum_{l=1}^{L} \sum_{i=1}^{N} (x_{ij} \ast \beta_i^l \ast y_i^l) \leq A_j^{rest}, \forall \ j. \tag{2}
\]
Flow Mode Selection and Switch Mapping problem

Flow programmability constraint

- $r$ is used to denote the least programmability of all offline flows

$$ pro^l = \sum_{i=1}^{N} \sum_{j=1}^{M} (\bar{p}_{ij}^l * x_{ij}^l * y_{ij}^l) \geq r, \forall l. $$ (3)

Propagation delay constraint

- Total propagation delay should not exceed the propagation delay of the ideal recovering case

$$ G = \sum_{j=1}^{M} \sum_{i=1}^{N} (\alpha_{ij} * \gamma_{ij} * D_{ij}). $$ (4)

$$ \sum_{l=1}^{L} \sum_{i=1}^{N} \sum_{j=1}^{M} (x_{ij}^l * \beta_{ij}^l * y_{ij}^l * D_{ij}) \leq G. $$ (5)
Flow Mode Selection and Switch Mapping problem

Flow programmability constraint

- \( r \) is used to denote the least programmability of all offline flows

\[
pro^l = \sum_{i=1}^{N} \sum_{j=1}^{M} (\bar{p}^l_{ij} \times x^l_{ij} \times y^l_{ij}) \geq r, \forall l.
\] (3)

Propagation delay constraint

- Total propagation delay should not exceed the propagation delay of the ideal recovering case

\[
G = \sum_{j=1}^{M} \sum_{i=1}^{N} (\alpha_{ij} \times \gamma_i \times D_{ij}).
\] (4)

\[
\sum_{l=1}^{L} \sum_{j=1}^{M} \sum_{i=1}^{N} (x^l_{ij} \times \beta^l_i \times y^l_i \times D_{ij}) \leq G.
\] (5)
Flow Mode Selection and Switch Mapping problem

Objective function

- To recover offline flows as many as possible and let them have similar programmability

\[ \text{obj}_1 = r. \] (6)

- To make full use of the control resource of active controllers

\[ \text{obj}_2 = \sum_{l=1}^{L} \text{pro}^l. \] (7)

Problem formulation

\[ \max_{r,x,y} r + \lambda \sum_{l=1}^{L} \text{pro}^l \] (P)

subject to

Eqs. (1)(2)(3)(4)(5)

\[ r \geq 0, \; x_{ij}, \; y^l_i \in \{0, 1\}, \; \forall \; i, \; \forall \; j, \; \forall \; l. \]
Flow Mode Selection and Switch Mapping problem

Objective function

- To recover offline flows as many as possible and let them have similar programmability
  \[ obj_1 = r. \] (6)

- To make full use of the control resource of active controllers
  \[ obj_2 = \sum_{l=1}^{L} pro^l. \] (7)

Problem formulation

\[
\begin{align*}
\max_{r,x,y} & \quad r + \lambda \sum_{l=1}^{L} pro^l \\
\text{subject to} & \quad Eqs. \ (1)(2)(3)(4)(5) \\
& \quad r \geq 0, \ x_{ij}, y_i^l \in \{0, 1\}, \forall \ i, \forall \ j, \forall \ l.
\end{align*}
\]
Outline

1. Background and Motivation
   1.1. Software-Defined Networking (SDN)
   1.2. Software-Defined Wide Area Networks (SD-WANs)
   1.3. Controller failure problem in SD-WANs

2. Existing Solutions and Limitations
   2.1. Switch-level mapping solution
   2.2. Flow-level mapping solution

3. Overview of ProgrammabilityMedic
   3.1. Opportunity
   3.2. Design considerations
   3.3. Flow Mode Selection and Switch Mapping problem
   3.4. Heuristic solution: ProgrammabilityMedic

4. Evaluation

5. Summary
Heuristic solution: ProgrammabilityMedic

Preferably recovering offline flows, which have the least programmability

- Finding switch $s_{i_0}$ to recover
- Mapping switch $s_{i_0}$ to controller $C_{j_0}$
- Selecting the routing mode for flows at switch $s_{i_0}$

Making full use of available control resource to improve the total programmability

- Improving the total programmability
Heuristic solution: ProgrammabilityMedic

Preferably recovering offline flows, which have the least programmability

- Finding switch $s_{i_0}$ to recover
- Mapping switch $s_{i_0}$ to controller $C_{j_0}$
- Selecting the routing mode for flows at switch $s_{i_0}$

Making full use of available control resource to improve the total programmability

- Improving the total programmability
1. Background and Motivation
   1.1. Software-Defined Networking (SDN)
   1.2. Software-Defined Wide Area Networks (SD-WANs)
   1.3. Controller failure problem in SD-WANs

2. Existing Solutions and Limitations
   2.1. Switch-level mapping solution
   2.2. Flow-level mapping solution

3. Overview of ProgrammabilityMedic
   3.1. Opportunity
   3.2. Design considerations
   3.3. Flow Mode Selection and Switch Mapping problem
   3.4. Heuristic solution: ProgrammabilityMedic

4. Evaluation

5. Summary
Evaluation

Simulation setup

- AT&T topology with 25 nodes and 112 (56*2) links
- 6 controllers
- Any two nodes have a flow

Comparison algorithms

- RetroFlow
- ProgrammabilityMedic
- Optimal
- ProgrammabilityGurdian

AT&T topology.
Evaluation

Simulation setup
- AT&T topology with 25 nodes and 112 (56*2) links
- 6 controllers
- Any two nodes have a flow

Comparison algorithms
- RetroFlow
- ProgrammabilityMedic
- Optimal
- ProgrammabilityGurdian

AT&T topology.
Evaluation

Simulation scenarios

- Scenario 1. One controller failure
- Scenario 2. Two controllers failure
- Scenario 3. Three controllers failure

Performance metrics

- Path programmability of flows
- Percentage of total path programmability of recovered flows to RetroFlow
- Percentage of recovered programmable flows from offline switches
- Number of recovered offline switches
- Control resource of active controllers
- Per-flow communication overhead
## Evaluation

### Two controllers failure

- **Recovered programmable flows**
  - PM, Optimal, and PG recover all offline flows
  - RetroFlow: 78-99% to PM

- **Communication overhead**
  - The lower, the better
  - PG performs the worst because the middle layer’s processing time significantly increases the overhead

---

### Recovered programmable flows

<table>
<thead>
<tr>
<th>ID of failed controllers</th>
<th>RetroFlow</th>
<th>PM</th>
<th>Optimal</th>
<th>PG</th>
</tr>
</thead>
<tbody>
<tr>
<td>[13,20]</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>75</td>
</tr>
<tr>
<td>[13,22]</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>65</td>
</tr>
<tr>
<td>[2,13]</td>
<td>80</td>
<td>80</td>
<td>80</td>
<td>50</td>
</tr>
<tr>
<td>[2,20]</td>
<td>70</td>
<td>70</td>
<td>70</td>
<td>40</td>
</tr>
<tr>
<td>[2,22]</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>30</td>
</tr>
<tr>
<td>[2,5]</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>20</td>
</tr>
<tr>
<td>[2,6]</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>10</td>
</tr>
<tr>
<td>[5,20]</td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>0</td>
</tr>
<tr>
<td>[5,22]</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>0</td>
</tr>
<tr>
<td>[5,13]</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>[5,20]</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

### Communication overhead per flow (ms)

<table>
<thead>
<tr>
<th>ID of failed controllers</th>
<th>RetroFlow</th>
<th>PM</th>
<th>Optimal</th>
<th>PG</th>
</tr>
</thead>
<tbody>
<tr>
<td>[13,20]</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>[13,22]</td>
<td>0.8</td>
<td>0.8</td>
<td>0.8</td>
<td>0.8</td>
</tr>
<tr>
<td>[2,13]</td>
<td>0.6</td>
<td>0.6</td>
<td>0.6</td>
<td>0.6</td>
</tr>
<tr>
<td>[2,20]</td>
<td>0.4</td>
<td>0.4</td>
<td>0.4</td>
<td>0.4</td>
</tr>
<tr>
<td>[2,22]</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>[2,5]</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>[2,6]</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>[5,20]</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>[5,22]</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>[5,13]</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>[5,20]</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>[5,22]</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>[5,13]</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>[5,20]</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>[5,22]</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>[5,13]</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>[5,20]</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>[5,22]</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>[5,13]</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>[5,20]</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>[5,22]</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>[5,13]</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>[5,20]</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>[5,22]</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>[5,13]</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

---

More results in paper
1. Background and Motivation
   1.1. Software-Defined Networking (SDN)
   1.2. Software-Defined Wide Area Networks (SD-WANs)
   1.3. Controller failure problem in SD-WANs

2. Existing Solutions and Limitations
   2.1. Switch-level mapping solution
   2.2. Flow-level mapping solution

3. Overview of ProgrammabilityMedic
   3.1. Opportunity
   3.2. Design considerations
   3.3. Flow Mode Selection and Switch Mapping problem
   3.4. Heuristic solution: ProgrammabilityMedic

4. Evaluation

5. Summary
Summary

New idea

- We propose to recover path programmability of offline flows by approximately realizing flow-controller mappings using **hybrid SDN/legacy routing** supported by high-end commercial SDN switches.

New problem and solution

- We formulate the **Flow Mode Selection and Switch Mapping (FMSSM) problem**, which is a mix integer programming with high computation complexity. To efficiently solve the problem, we reformulate the problem with linear techniques and solve the problem with the proposed efficient **heuristic algorithm** named PM.

Good performance

- We evaluate the performance of PM under different controller failure scenarios. The results show that PM outperforms existing switch-level solutions by maintaining balanced programmability and increasing the total programmability under multiple controller failures.
Summary

New idea

• We propose to recover path programmability of offline flows by approximately realizing flow-controller mappings using hybrid SDN/legacy routing supported by high-end commercial SDN switches.

New problem and solution

• We formulate the Flow Mode Selection and Switch Mapping (FMSSM) problem, which is a mix integer programming with high computation complexity. To efficiently solve the problem, we reformulate the problem with linear techniques and solve the problem with the proposed efficient heuristic algorithm named PM.

Good performance

• We evaluate the performance of PM under different controller failure scenarios. The results show that PM outperforms existing switch-level solutions by maintaining balanced programmability and increasing the total programmability under multiple controller failures.
New idea

• We propose to recover path programmability of offline flows by approximately realizing flow-controller mappings using hybrid SDN/legacy routing supported by high-end commercial SDN switches.

New problem and solution

• We formulate the Flow Mode Selection and Switch Mapping (FMSSM) problem, which is a mix integer programming with high computation complexity. To efficiently solve the problem, we reformulate the problem with linear techniques and solve the problem with the proposed efficient heuristic algorithm named PM.

Good performance

• We evaluate the performance of PM under different controller failure scenarios. The results show that PM outperforms existing switch-level solutions by maintaining balanced programmability and increasing the total programmability under multiple controller failures.
Thank you for your attention!

Q&A

songshidou@hotmail.com