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)

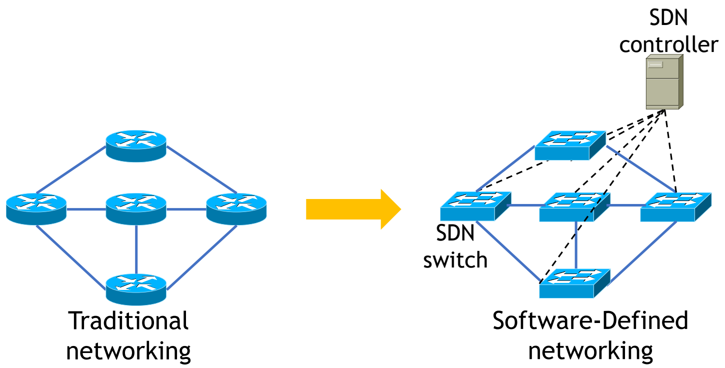
Outline

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

Outline

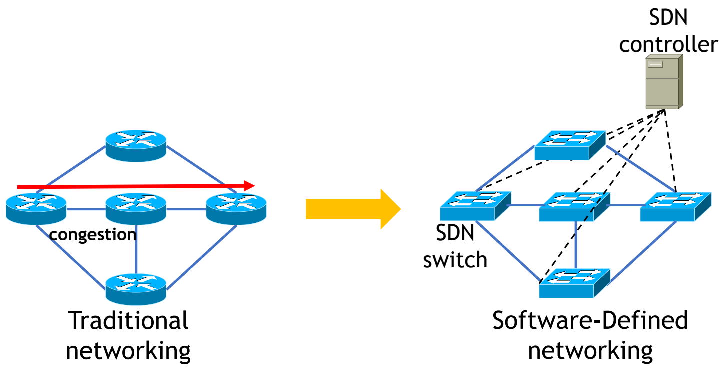
Software-Defined Networking (SDN)

北京理工大学 BEIJING INSTITUTE OF TECHNOLOGY



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

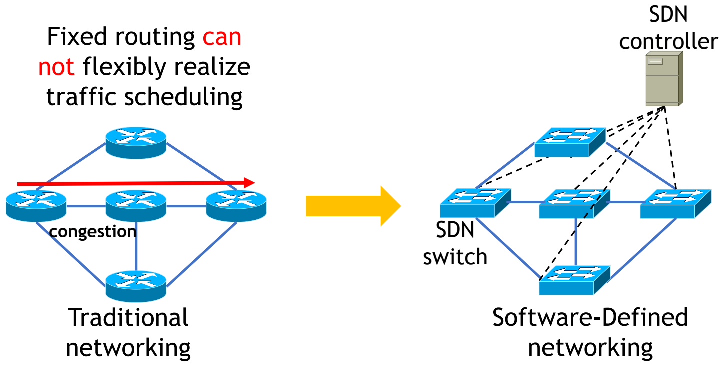
Software-Defined Networking (SDN)



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

Software-Defined Networking (SDN)

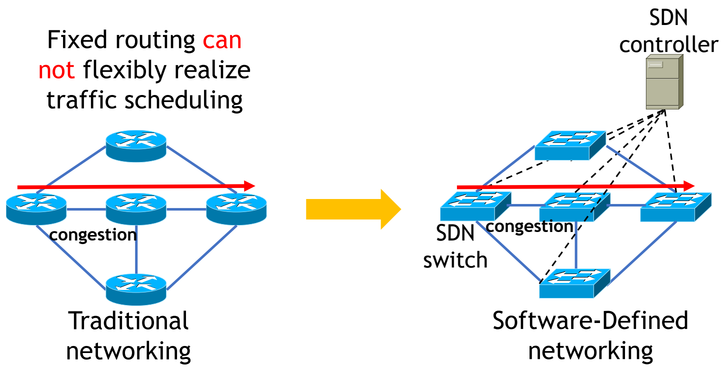
北京理工大學
BEIJING INSTITUTE OF TECHNOLOGY



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

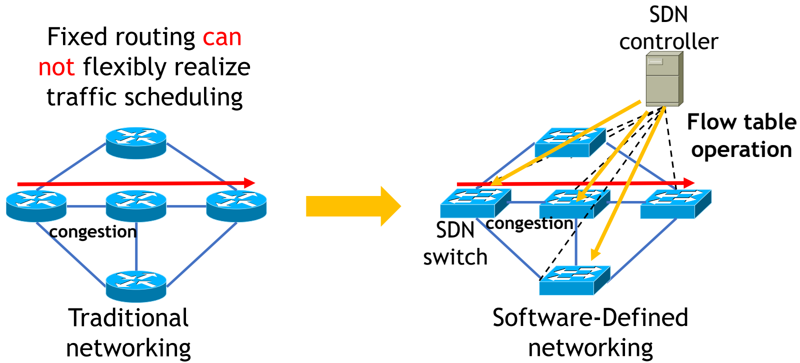
Software-Defined Networking (SDN)

北京理工大学 BEIJING INSTITUTE OF TECHNOLOGY



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

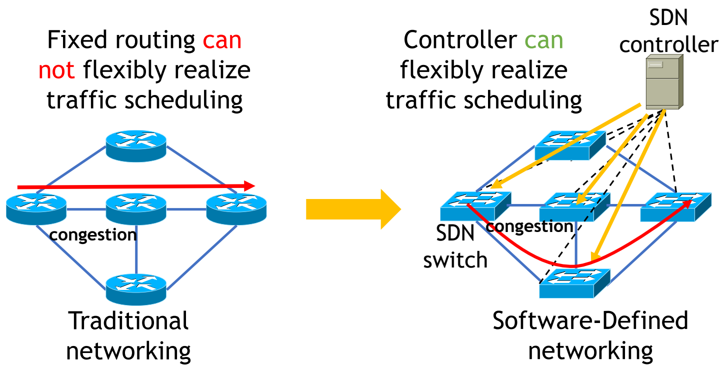
Software-Defined Networking (SDN)



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

Software-Defined Networking (SDN)

北京理工大学 BEIJING INSTITUTE OF TECHNOLOGY

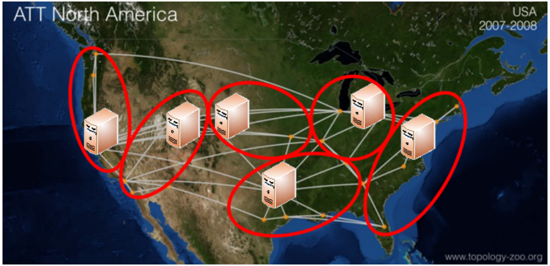


Advantages of SDN (e.g., path programmability).

Outline

Software-Defined Wide Area Networks (SD-WANs)

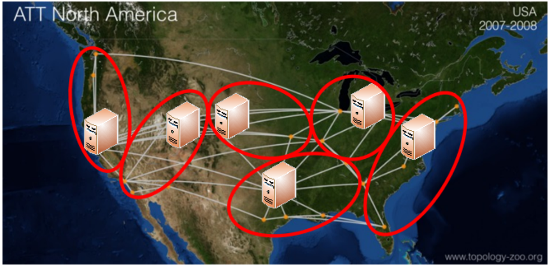
- Large scale with many devices
- Partitioning the network into domains
- Distributed control plane
  1. Quick response
  2. Control resiliency
  3. Controller synchronization



AT&T topology.

Software-Defined Wide Area Networks (SD-WANs)

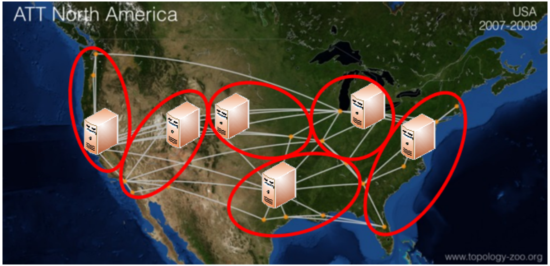
- Large scale with many devices
- Partitioning the network into domains
- Distributed control plane
  1. Quick response
  2. Control resiliency
  3. Controller synchronization



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.

Outline

Controller failure problem in SD-WANs

北京理工大学 BEIJING INSTITUTE OF TECHNOLOGY

Controller failure

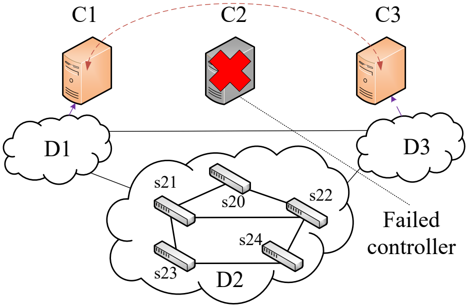
- Software bugs, Hardware failure, Power outage

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

北京理工大学 BEIJING INSTITUTE OF TECHNOLOGY

Controller failure

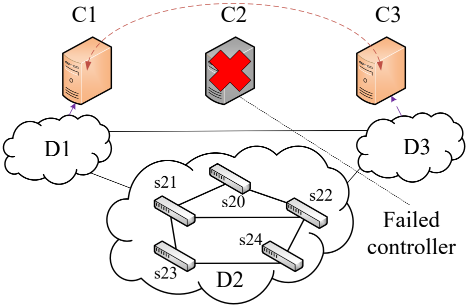
- Software bugs, Hardware failure, Power outage

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

北京理工大学 BEIJING INSTITUTE OF TECHNOLOGY

Controller failure

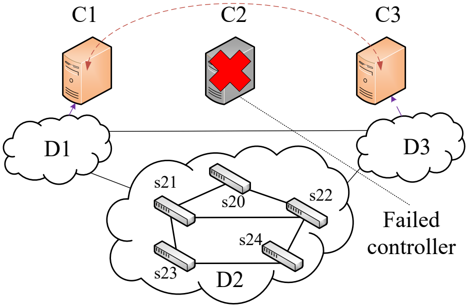
- Software bugs, Hardware failure, Power outage

Maintaining path programmability

- Offline switches remapping

Challenges

- Limited control resource
- Path programmability recovery
- Multiple failures



An example of controller failure.

Outline

北京理工大学 BEIJING INSTITUTE OF TECHNOLOGY

Outline

北京理工大学
BEIJING INSTITUTE OF TECHNOLOGY

Switch-level mapping solution

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

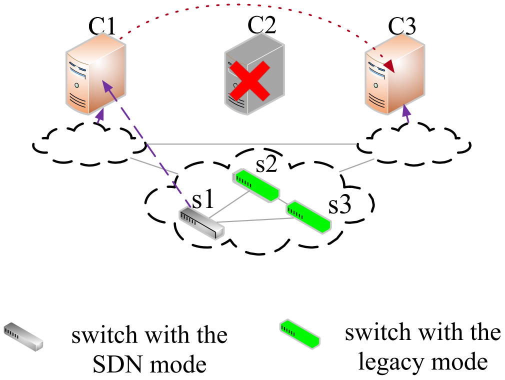
Switch-level mapping solution

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



An example of switch-level solution.

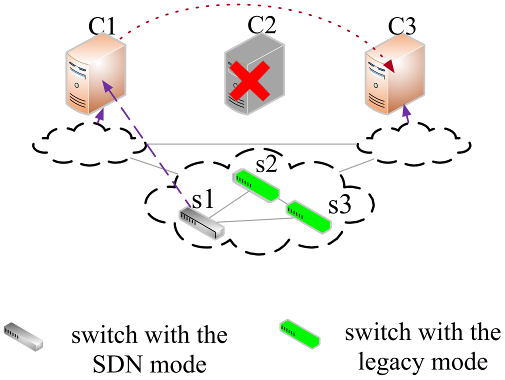
*Z. Guo et al., Retroflow: Maintaining control resiliency and flow programmability for software-defined wans, in IEEE/ACM IWQoS'19.

Switch-level mapping solution

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



An example of switch-level solution.

Limitations

- Fixed control cost of switches
- Per-switch mapping

*Z. Guo et al., Retroflow: Maintaining control resiliency and flow programmability for software-defined wans, in IEEE/ACM IWQoS'19.

Outline

BEIJING INSTITUTE OF TECHNOLOGY

Flow-level mapping solution

北京理工大学 BEIJING INSTITUTE OF TECHNOLOGY

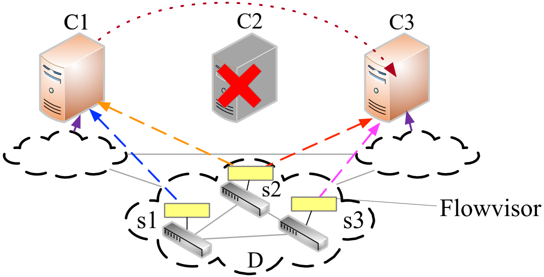
Switch-level mapping solution

ProgrammabilityGuardian†

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

Limitations

- Increasing the processing delay
- Bringing new unreliability



An example of flow-level solution.

†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

北京理工大学 BEIJING INSTITUTE OF TECHNOLOGY

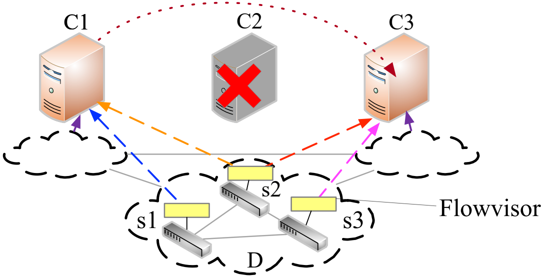
Switch-level mapping solution

ProgrammabilityGuardian†

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

Limitations

- Increasing the processing delay
- Bringing new unreliability



An example of flow-level solution.

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

Outline

BEIJING INSTITUTE OF TECHNOLOGY

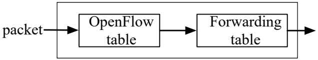
Outline

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

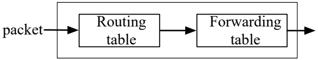
Opportunity

Hybrid SDN/legacy routing mode

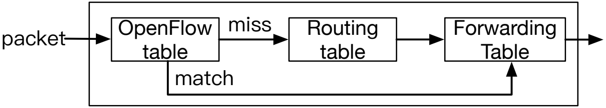
- Specifically hybrid OpenFlow/OSPF routing mode
- Selectively deciding the routing mode for each offline flows
- Dynamically changing the control cost of offline flows



SDN routing mode.



Legacy routing mode.



Hybrid SDN/legacy routing mode.

Outline

Design considerations

北京理工大學
BEIJING INSTITUTE OF TECHNOLOGY

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

北京理工大学
BEIJING INSTITUTE OF TECHNOLOGY

Consideration 1:

- Recovering offline flows as many as possible

Consideration 2:

- Balancing path programmability

Consideration 3:

- Fully utilizing controllers' control resource

Outline

Flow Mode Selection and Switch Mapping problem

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

Switch-controller mapping constraint

- Each switch can be mapped to at most one controller

$$\sum_{j=1}^{M} x_{ij} \leq 1, \forall\ i. \quad (1)$$

Controller resource constraint

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

$$\sum_{l=1}^{L} \sum_{i=1}^{N} (x_{ij} * \beta_i^l * y_i^l) \leq A_j^{rest}, \forall\ j. \quad (2)$$

Flow Mode Selection and Switch Mapping problem

北京理工大学 BEIJING INSTITUTE OF TECHNOLOGY

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

$$\sum_{l=1}^{L}\sum_{i=1}^{N}(x_{ij} * \beta_i^l * y_i^l) \leq A_j^{rest}, \forall\ j. \tag{2}$$

Flow Mode Selection and Switch Mapping problem

北京理工大学 BEIJING INSTITUTE OF TECHNOLOGY

Flow programmability constraint

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

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

Propagation delay constraint

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

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

$$\sum_{l=1}^{L} \sum_{j=1}^{M} \sum_{i=1}^{N} (x_{ij} * \beta_i^l * y_i^l * D_{ij}) \leq G. \quad (5)$$

Flow programmability constraint

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

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

Propagation delay constraint

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

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

$$\sum_{l=1}^{L} \sum_{j=1}^{M} \sum_{i=1}^{N} (x_{ij} * \beta_i^l * y_i^l * D_{ij}) \leq G. \tag{5}$$

Flow Mode Selection and Switch Mapping problem

北京理工大学 BEIJING INSTITUTE OF TECHNOLOGY

Objective function

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

$$obj_1 = r. \tag{6}$$

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

$$obj_2 = \sum_{l=1}^{L} pro^l. \tag{7}$$

Problem formulation

$$\max_{r,x,y} r + \lambda \sum_{l=1}^{L} pro^l \tag{P}$$

subject to

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

$$r \geq 0, x_{ij}, y_i^l \in \{0,1\}, \forall\, i, \forall\, j, \forall\, l.$$

Objective function

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

$$obj_1 = r. \tag{6}$$

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

$$obj_2 = \sum_{l=1}^{L} pro^l. \tag{7}$$

Problem formulation

$$\max_{r,x,y} r + \lambda \sum_{l=1}^{L} pro^l \tag{P}$$

subject to

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

$$r \geq 0, x_{ij}, y_i^l \in \{0,1\}, \forall\, i, \forall\, j, \forall\, l.$$

Outline

北京理工大学 BEIJING INSTITUTE OF TECHNOLOGY

Heuristic solution: ProgrammabilityMedic

北京理工大学
BEIJING INSTITUTE OF TECHNOLOGY

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

```
X = ∅, Y = ∅, S* = S, σ = 0, test_count = 0;
while test_count ≤ TOTAL_ITERATIONS do
    δ = 0, i_0 = NULL, j_0 = NULL;
    //find switch s_{i_0} to recover
    for s_i ∈ S* do
        TEST_NUM = 0;
        for l ∈ {β_i^l = 1, l ∈ [1, L]} do
            if h^l == σ then
                TEST_NUM = TEST_NUM + 1;
            end if
        end for
        if TEST_NUM > δ then
            δ = TEST_NUM, i_0 = i;
        end if
    end for
    //map switch s_{i_0} to controller C_{j_0}
    if (i_0, *) ∈ X then
        use i_0 to find j_0 from X;
    else
        for C_j ∈ C(i_0) do
            if A_j^{rest} ≥ γ_{i_0} then
                j_0 = j;
            end if
        end for
        if j_0 == NULL then
            A_j^{rest} = max(A), j_0 = j;
        end if
    end if
    X ← X ∪ (i_0, j_0), S* ← S* \ s_{i_0};
    //select the routing mode for flows at switch s_{i_0}
    for l_0 ∈ {β_{i_0}^l = 1, l ∈ [1, L]} do
        if h^{l_0} ≤ σ and (i_0, l_0) ∉ Y and A_{j_0}^{rest} > 0 then
            A_{j_0}^{rest} = A_{j_0}^{rest} − 1, h^{l_0} = h^{l_0} + pro_{i_0}^{l_0};
            Y ← Y ∪ (i_0, l_0);
        end if
    end for
    if |S*| == ∅ then
        S* = S, test_count++, σ = min(H);
    end if
end while
//improve the total programmability
for (i_0, l_0) ∈ {β_i^l = 1, i ∈ [1, M], l ∈ [1, L]} do
    if (i_0, *) ∈ X then
        use i_0 to find j_0 from X;
        if A_{j_0}^{rest} > 0 and (i_0, l_0) ∉ Y then
            A_{j_0}^{rest} = A_{j_0}^{rest} − 1, h^{l_0} = h^{l_0} + pro_{i_0}^{l_0};
            Y ← Y ∪ (i_0, l_0);
        end if
    end if
end for
return X, Y;
```

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

```
X = ∅, Y = ∅, S* = S, σ = 0, test_count = 0;
while test_count ≤ TOTAL_ITERATIONS do
    δ = 0, i_0 = NULL, j_0 = NULL;
    //find switch s_{i_0} to recover
    for s_i ∈ S* do
        TEST_NUM = 0;
        for l ∈ {β_i^l = 1, l ∈ [1, L]} do
            if h^l == σ then
                TEST_NUM = TEST_NUM + 1;
            end if
        end for
        if TEST_NUM > δ then
            δ = TEST_NUM, i_0 = i;
        end if
    end for
    //map switch s_{i_0} to controller C_{j_0}
    if (i_0, *) ∈ X then
        use i_0 to find j_0 from X;
    else
        for C_j ∈ C(i_0) do
            if A_j^{rest} ≥ γ_{i_0} then
                j_0 = j;
            end if
        end for
        if j_0 == NULL then
            A_j^{rest} = max(A), j_0 = j;
        end if
    end if
    X ← X ∪ (i_0, j_0), S* ← S* \ s_{i_0};
    //select the routing mode for flows at switch s_{i_0}
    for l_0 ∈ {β_{i_0}^l = 1, l ∈ [1, L]} do
        if h^{l_0} ≤ σ and (i_0, l_0) ∉ Y and A_{j_0}^{rest} > 0 then
            A_{j_0}^{rest} = A_{j_0}^{rest} − 1, h^{l_0} = h^{l_0} + pro_{i_0}^{l_0};
            Y ← Y ∪ (i_0, l_0);
        end if
    end for
    if |S*| == ∅ then
        S* = S, test_count++, σ = min(H);
    end if
end while
//improve the total programmability
for (i_0, l_0) ∈ {β_i^l = 1, i ∈ [1, M], l ∈ [1, L]} do
    if (i_0, *) ∈ X then
        use i_0 to find j_0 from X;
        if A_{j_0}^{rest} > 0 and (i_0, l_0) ∉ Y then
            A_{j_0}^{rest} = A_{j_0}^{rest} − 1, h^{l_0} = h^{l_0} + pro_{i_0}^{l_0};
            Y ← Y ∪ (i_0, l_0);
        end if
    end if
end for
return X, Y;
```

Outline

北京理工大學
BEIJING INSTITUTE OF TECHNOLOGY

Evaluation

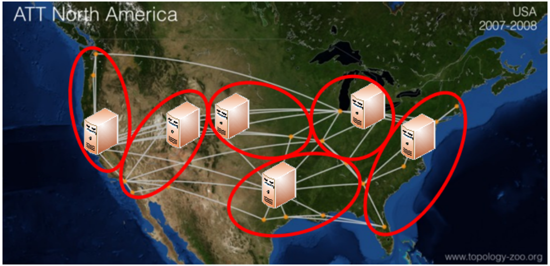
北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

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

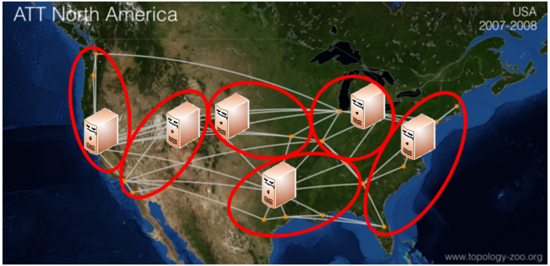
BEIJING INSTITUTE OF TECHNOLOGY

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

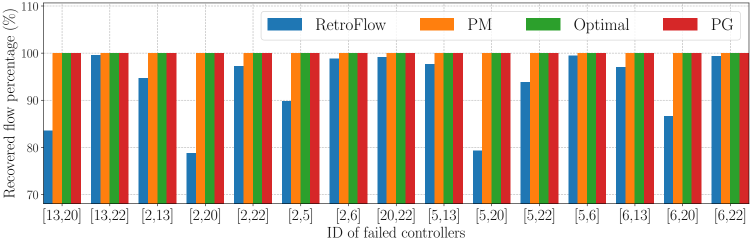
- **Path programmability** of flows
- Percentage of **total path programmability** of recovered flows to RetroFlow
- Percentage of **recovered programmable flows** from offline switches
- Number of **recovered offline switches**
- **Control resource** of active controllers
- Per-flow **communication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

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

Two controllers failure

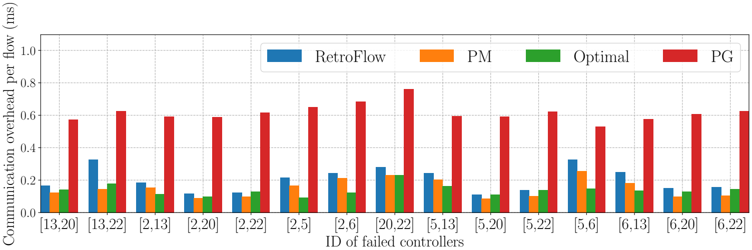
Recovered programmable flows

- 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



More results in paper

Outline

北京理工大學
BEIJING INSTITUTE OF TECHNOLOGY

Summary

北京理工大學
BEIJING INSTITUTE OF TECHNOLOGY

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

北京理工大学 BEIJING INSTITUTE OF TECHNOLOGY

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

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

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