Applied Networking Research Workshop 2020

Toward Programmable Interdomain Routing

Qiao Xiang[1], J. Jensen Zhang[1, 2], Franck Le[3], Y. Richard Yang[4, 1]

[1] Yale University, [2] Tongji University, [3] IBM T.J. Watson Research Center, [4] Peng Cheng Laboratory,

07/30/2020, ACM/IRTF ANRW'20

Background: What is Interdomain Routing

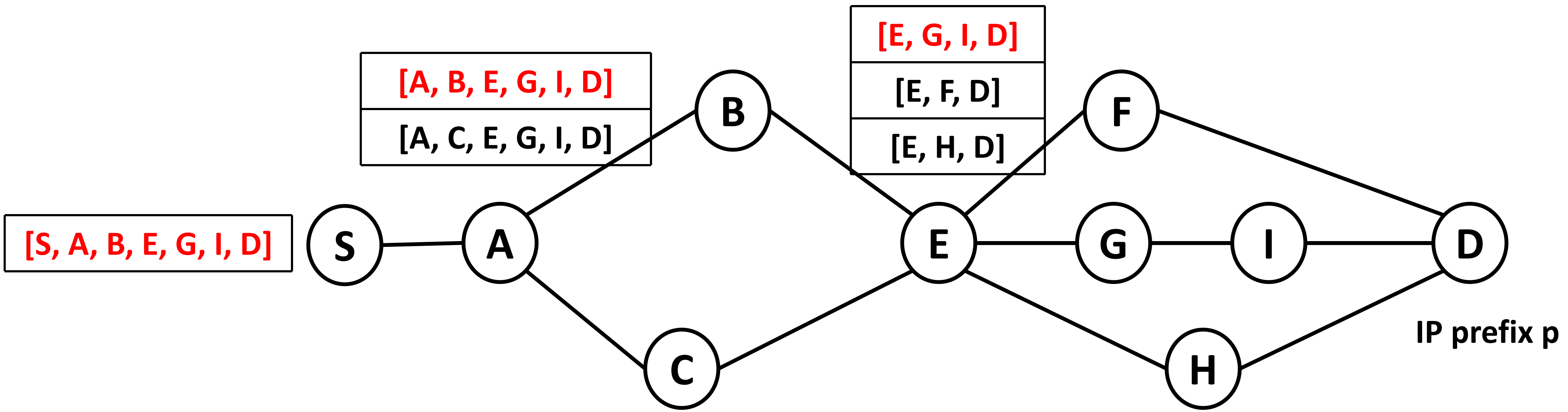
- Determine routes for source-destination pairs that span multiple ASes
    - Ideally, allow policy-routing, flexible traffic engineering and etc.
- *De facto* interdomain routing protocol: Border Gateway Protocol (BGP)

Background: What is Interdomain Routing

- Determine routes for source-destination pairs that span multiple ASes
  - Ideally, allow policy-routing, flexible traffic engineering and etc.
- *De facto* interdomain routing protocol: Border Gateway Protocol (BGP)
- **BGP in a nutshell**: Each AS makes and executes its own policy to select routes and export the selected routes in terms of path vectors (i.e., AS path), to its neighbor ASes

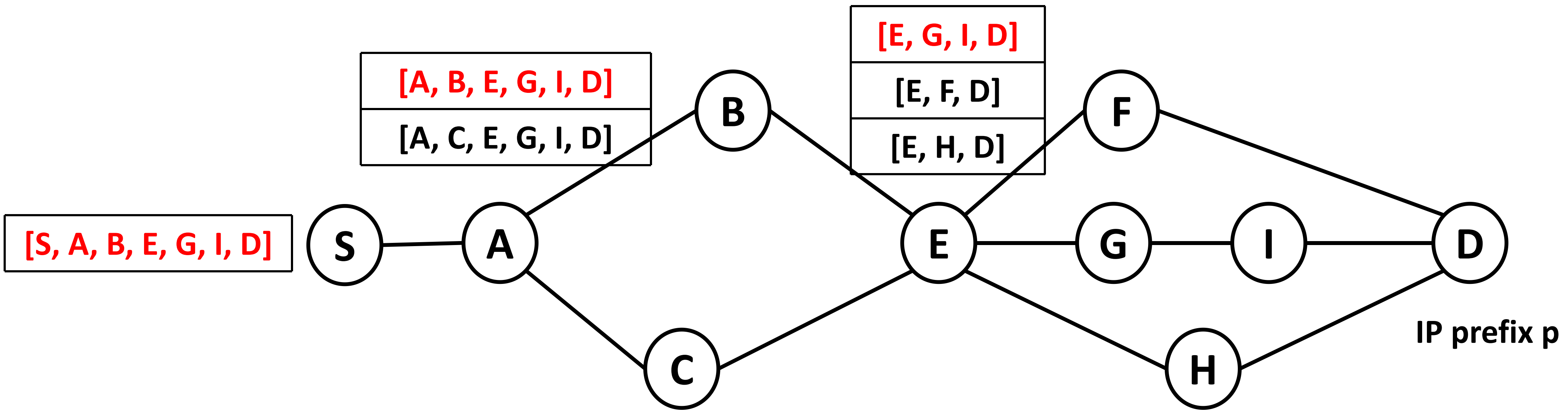
Background: What is Interdomain Routing

- Determine routes for source-destination pairs that span multiple ASes
  - Ideally, allow policy-routing, flexible traffic engineering and etc.
- *De facto* interdomain routing protocol: Border Gateway Protocol (BGP)
- **BGP in a nutshell**: Each AS makes and executes its own policy to select routes and export the selected routes in terms of path vectors (i.e., AS path), to its neighbor ASes

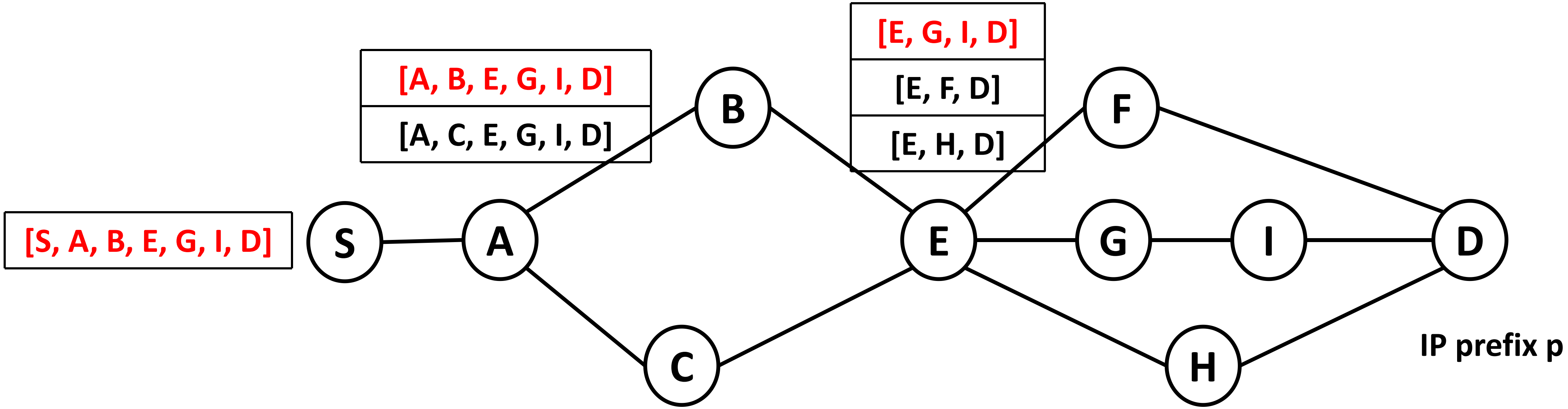


Background: What is Interdomain Routing

- Determine routes for source-destination pairs that span multiple ASes
  - Ideally, allow policy-routing, flexible traffic engineering and etc.
- *De facto* interdomain routing protocol: Border Gateway Protocol (BGP)
- **BGP in a nutshell**: Each AS makes and executes its own policy to select routes and export the selected routes in terms of path vectors (i.e., AS path), to its neighbor ASes
- BGP can implement policy-routing, but not other use cases such as flexible traffic engineering

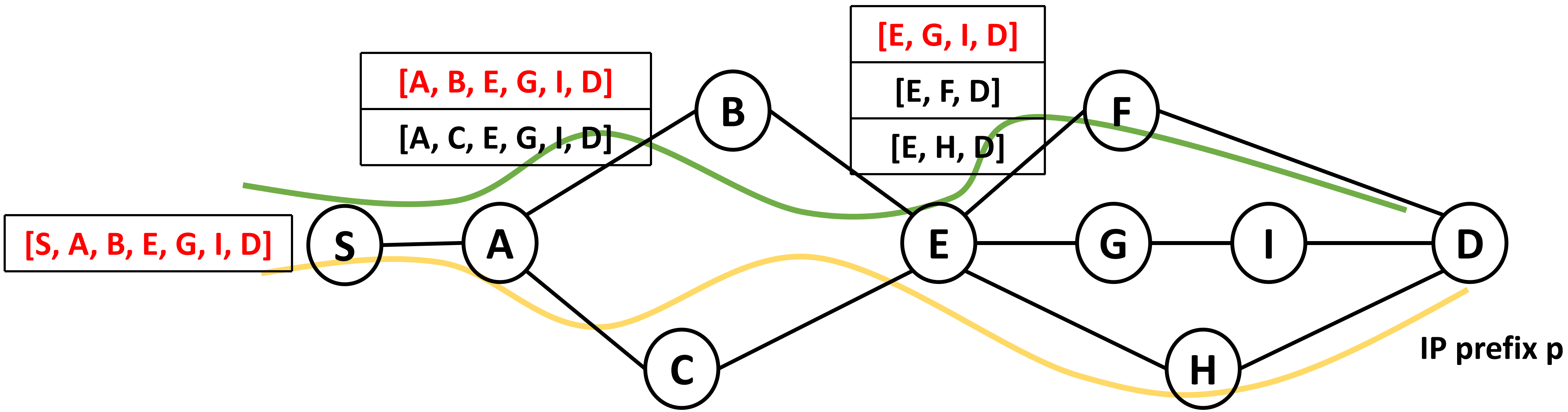


Limitation of BGP: Lacking Mechanisms for Flexible End-to-End Interdomain Route Control



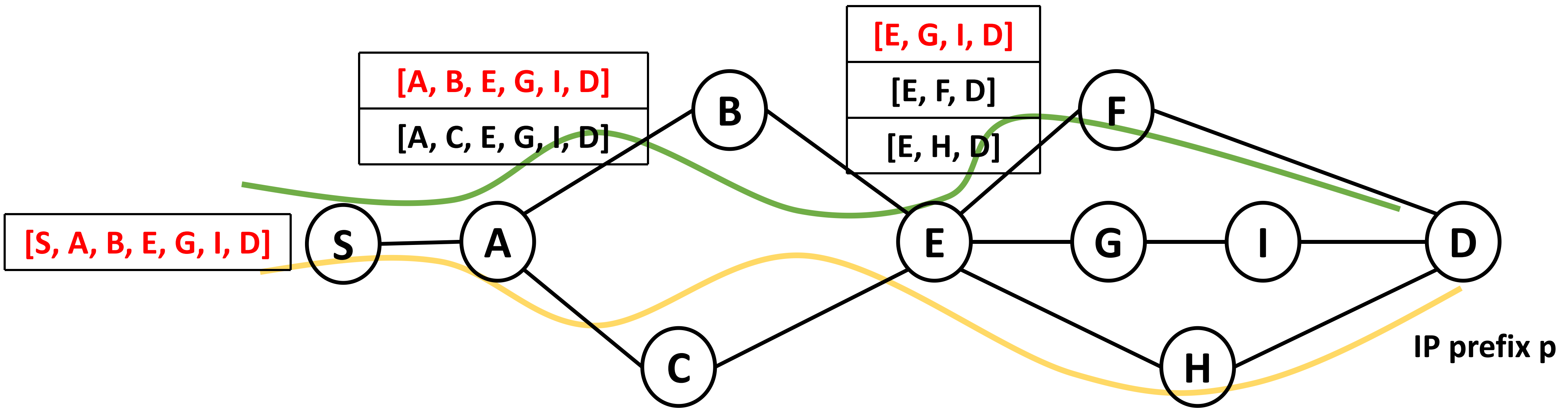
Limitation of BGP: Lacking Mechanisms for Flexible End-to-End Interdomain Route Control

- **Example**: Shorter AS-paths can be achieved, but S cannot select them



Limitation of BGP: Lacking Mechanisms for Flexible End-to-End Interdomain Route Control

- **Example**: Shorter AS-paths can be achieved, but S cannot select them
  - BGP does not provide mechanisms for S to control E's route selection



Goal of This Paper

Goal of This Paper

- A systematic formulation of the software-defined internetworking (SDI) model, extending intradomain SDN to generic interdomain SDN to support flexible, end-to-end interdomain route control

Goal of This Paper

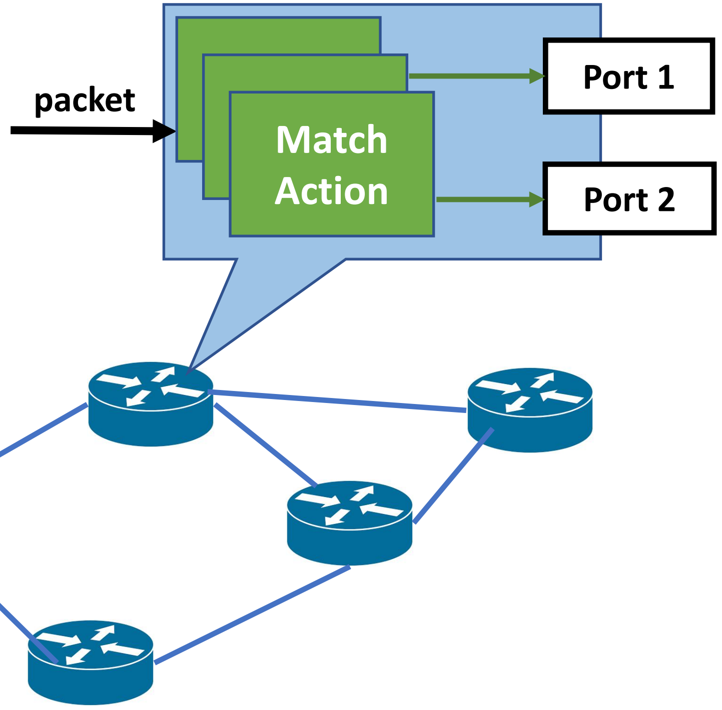
- A systematic formulation of the software-defined internetworking (SDI) model, extending intradomain SDN to generic interdomain SDN to support flexible, end-to-end interdomain route control
  - Conceptually program every single packet end-to-end in an interdomain network
  - Save users from the trouble of configuring and reasoning low-level details of interdomain routing (e.g., AS-path prepending, offline negotiation with different ASes and tunnel management)

Outline

- Introduction
- SDI network control model

Control Model: Abstract AS as Virtual Switch

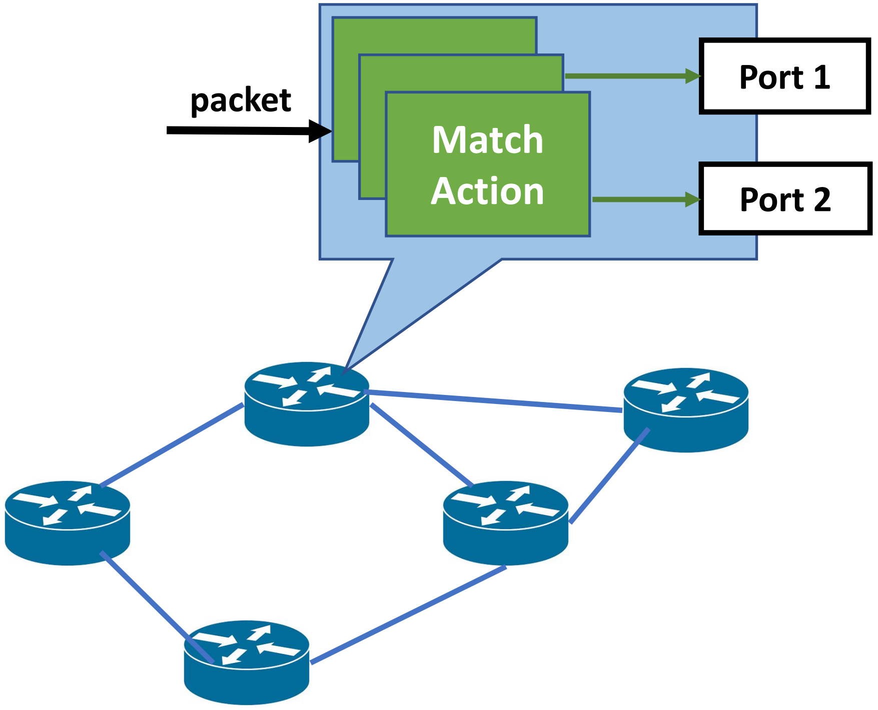
- Single domain SDN is very well understood, SDI aims to achieve similar things in interdomain setting



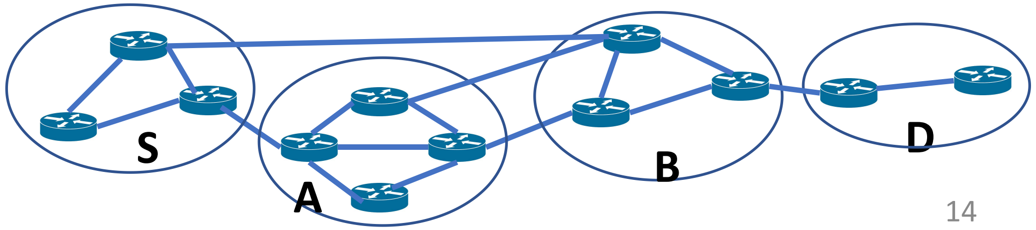
Single domain SDN

Control Model: Abstract AS as Virtual Switch

- Single domain SDN is very well understood, SDI aims to achieve similar things in interdomain setting



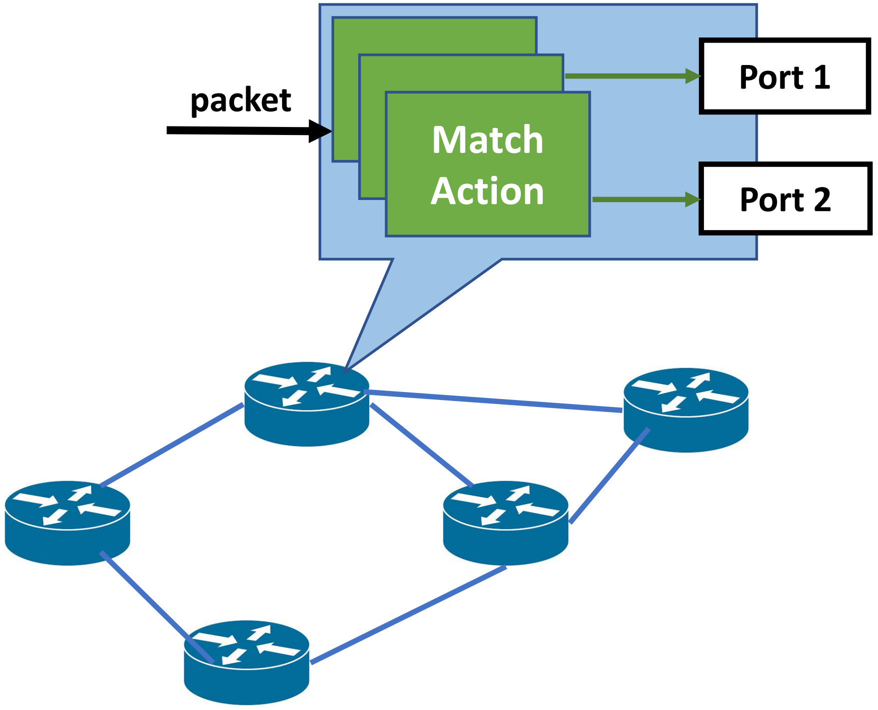
Single domain SDN



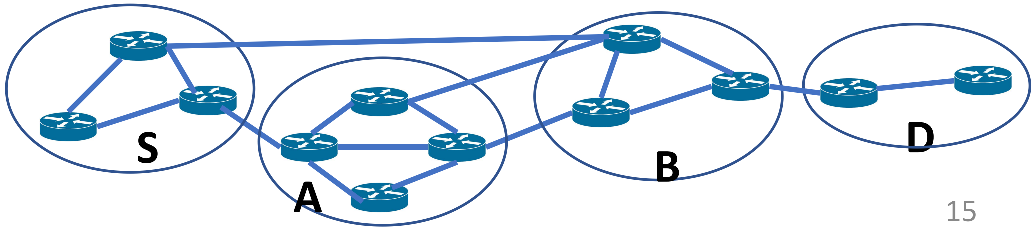
Interdomain Network

Control Model: Abstract AS as Virtual Switch

- Single domain SDN is very well understood, SDI aims to achieve similar things in interdomain setting
  - Each AS abstracted as a virtual switch with a pipeline of match-action tables and path-ports (i.e., AS paths), and exposed through north-bound protocol (e.g., ALTO)



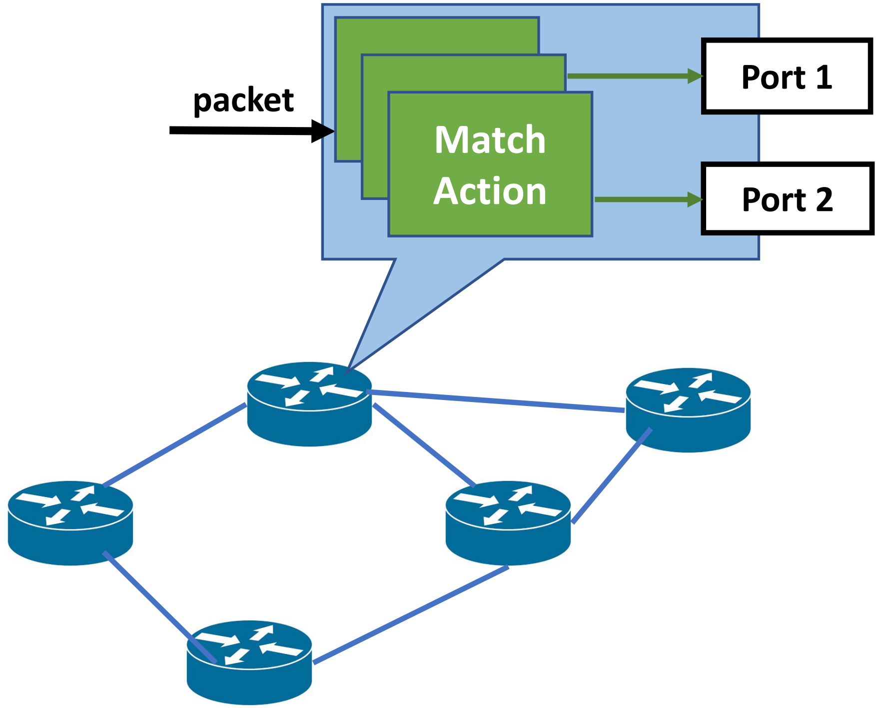
Single domain SDN



Interdomain Network

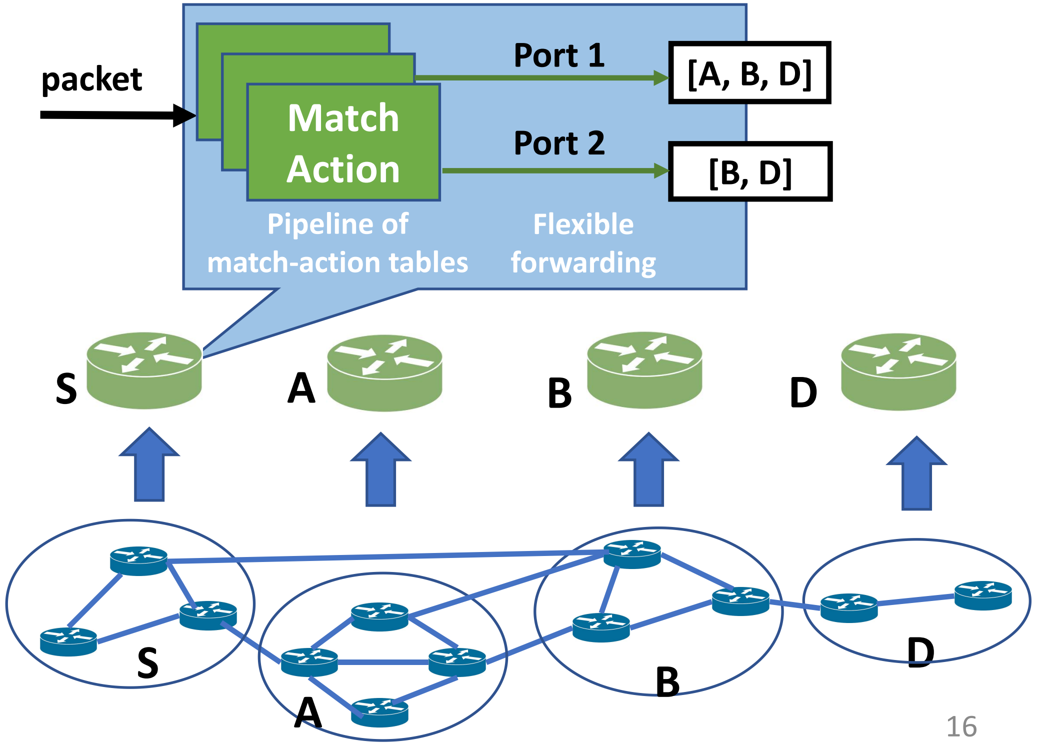
Control Model: Abstract AS as Virtual Switch

- Single domain SDN is very well understood, SDI aims to achieve similar things in interdomain setting
  - Each AS abstracted as a virtual switch with a pipeline of match-action tables and path-ports (i.e., AS paths), and exposed through north-bound protocol (e.g., ALTO)



Single domain SDN

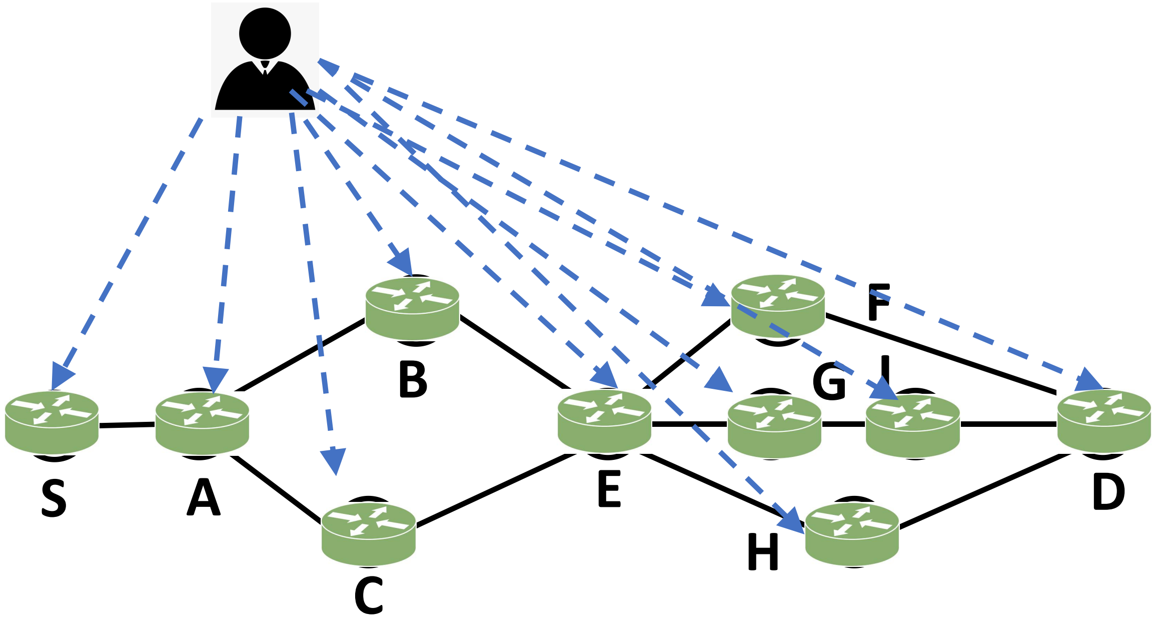
Each AS becomes an SDI-net



Interdomain Network

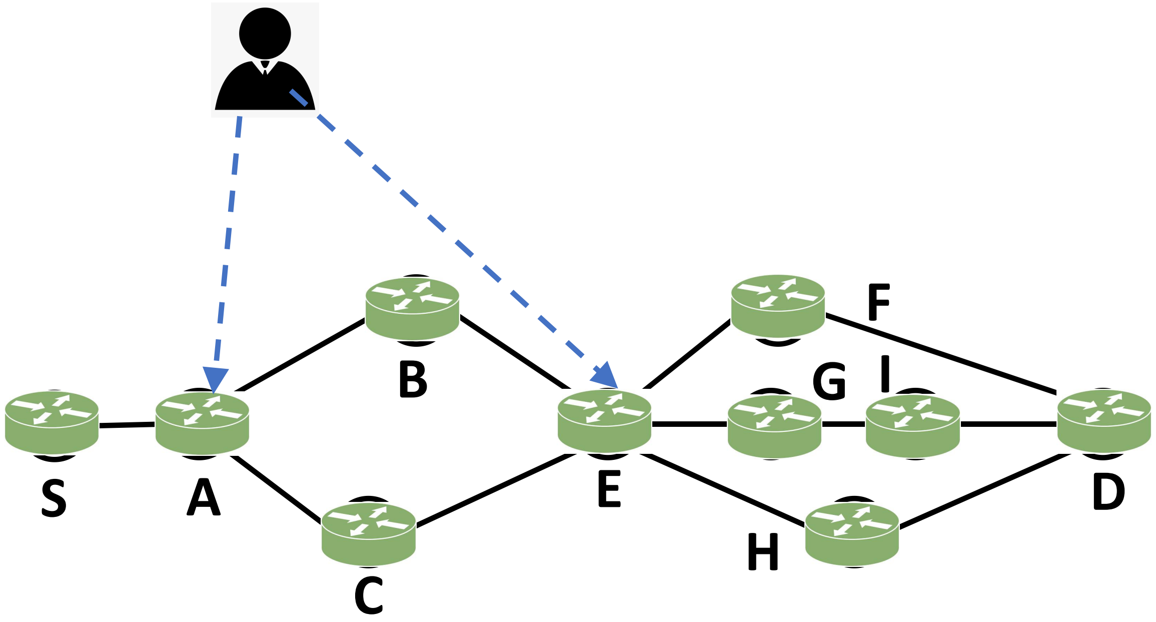
Control Model: Control SDI Network

- A client connects to SDI-nets to control paths in interdomain network



Control Model: Control SDI Network

- A client connects to SDI-nets to control paths in interdomain network
- A client may select to control a subset of SDI-nets to simplify management and business arrangements

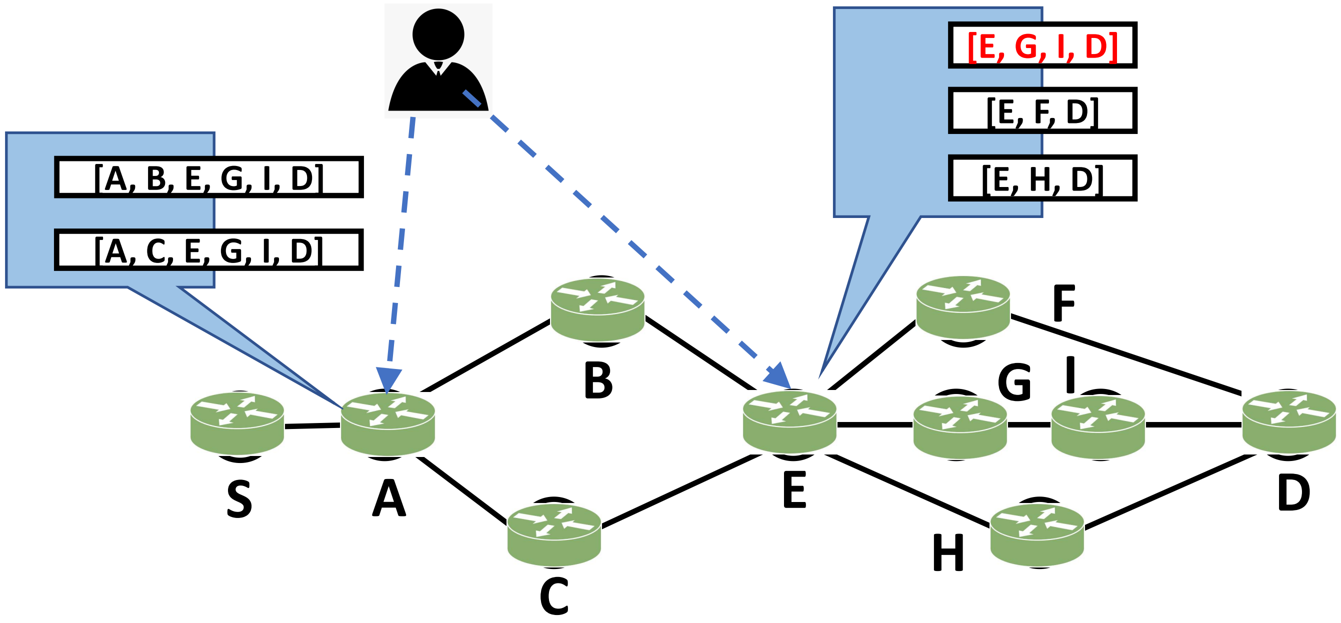


SDI vs. SDN: Key Difference

- Dynamic and dependent path-ports in SDI-nets
    - Upstream path-ports depend on downstream path-ports

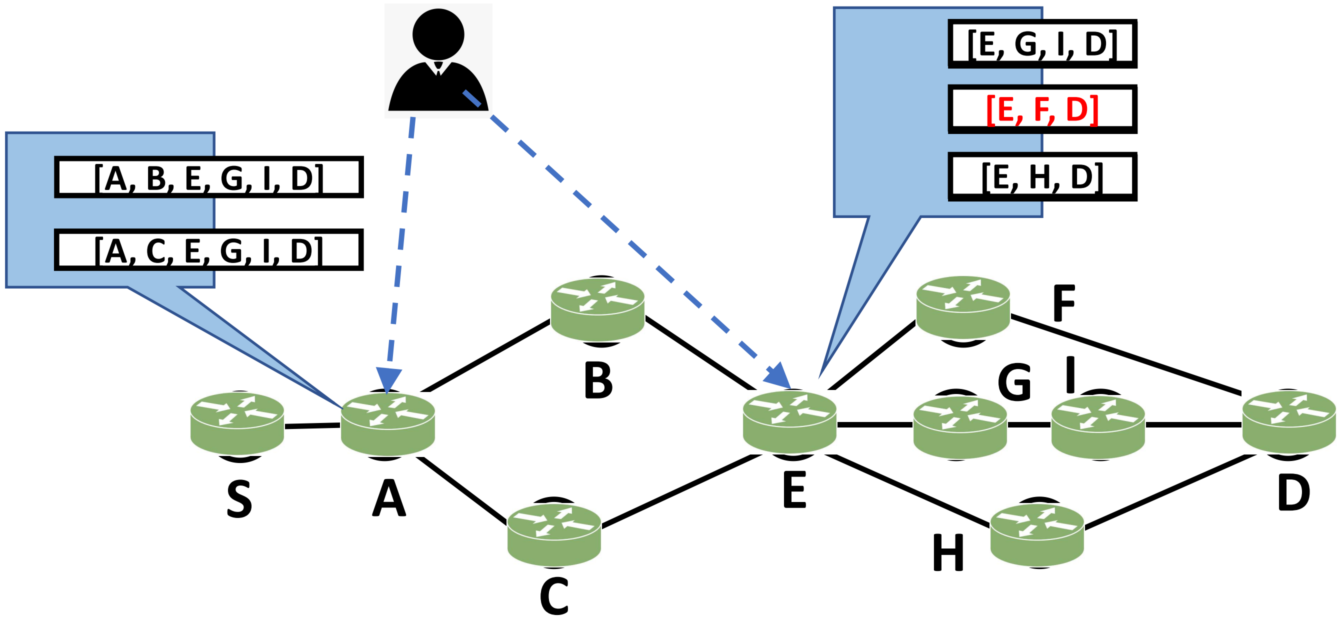
SDI vs. SDN: Key Difference

- Dynamic and dependent path-ports in SDI-nets
  - Upstream path-ports depend on downstream path-ports



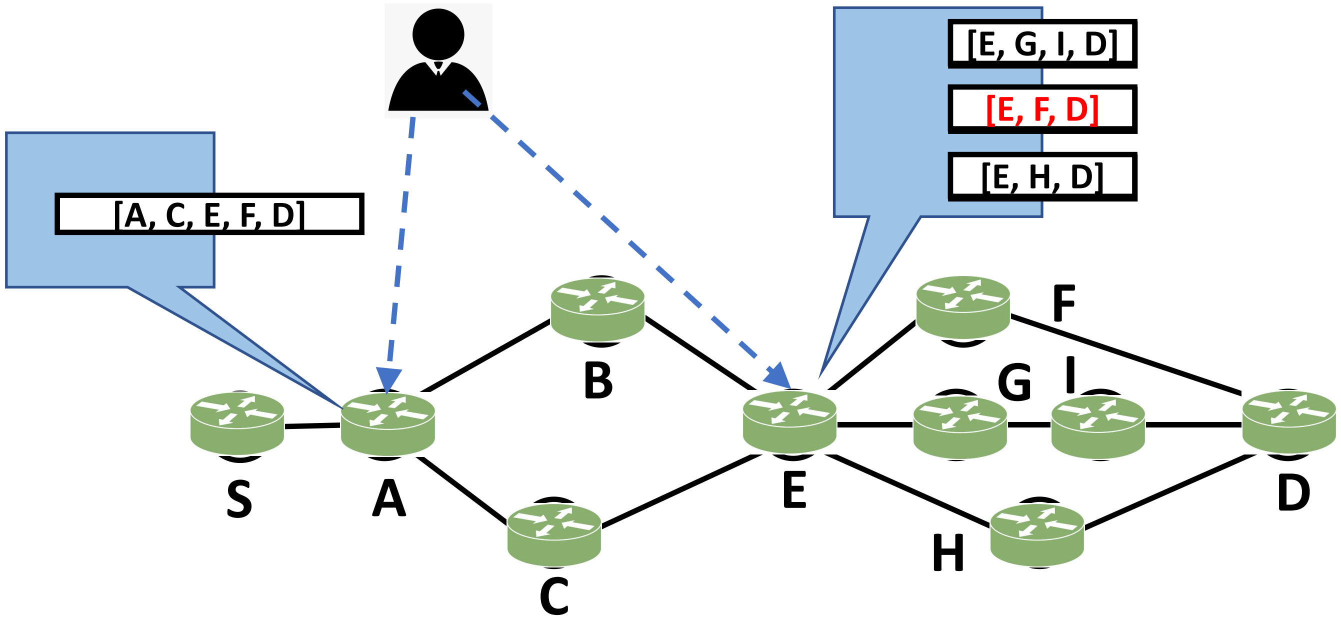
SDI vs. SDN: Key Difference

- Dynamic and dependent path-ports in SDI-nets
  - Upstream path-ports depend on downstream path-ports
- **Example**: When E selects [E, F, D] and does not export to B, A's path-port would become [A, C, E, F, D]



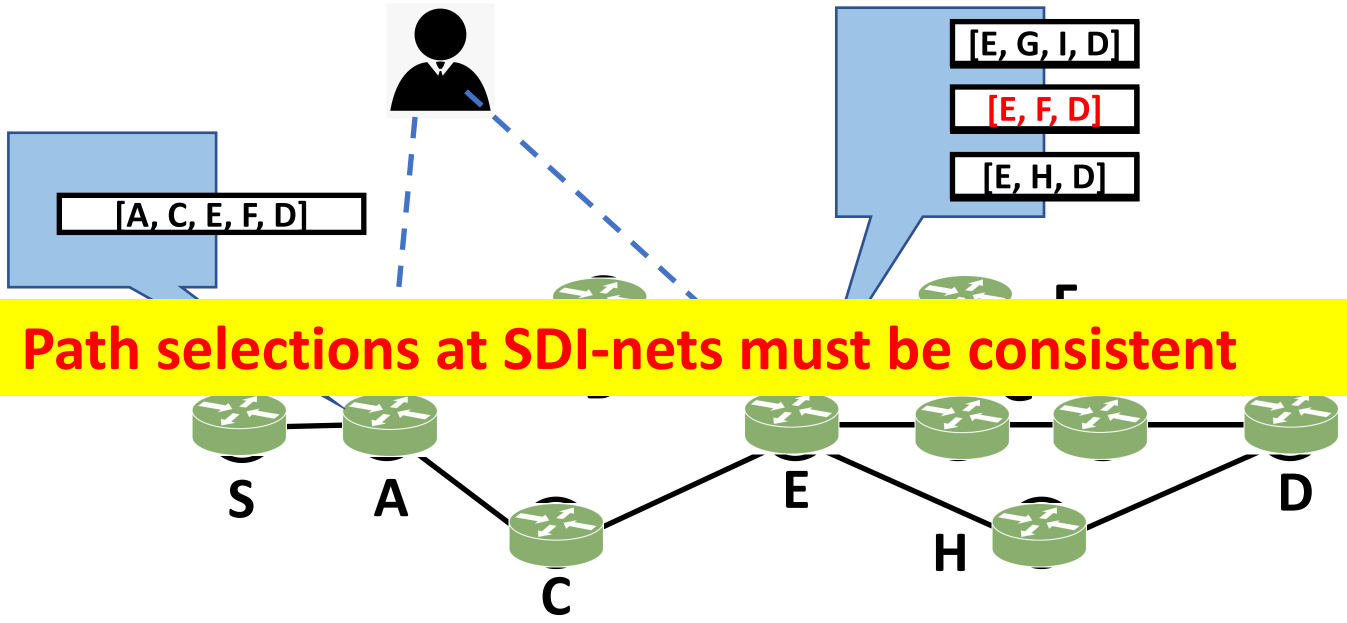
SDI vs. SDN: Key Difference

- Dynamic and dependent path-ports in SDI-nets
  - Upstream path-ports depend on downstream path-ports
- **Example**: When E selects [E, F, D] and does not export to B, A's path-port would become [A, C, E, F, D]



SDI vs. SDN: Key Difference

- Dynamic and dependent path-ports in SDI-nets
  - Upstream path-ports depend on downstream path-ports
- **Example**: When E selects [E, F, D] and does not export to B, A's path-port would become [A, C, E, F, D]



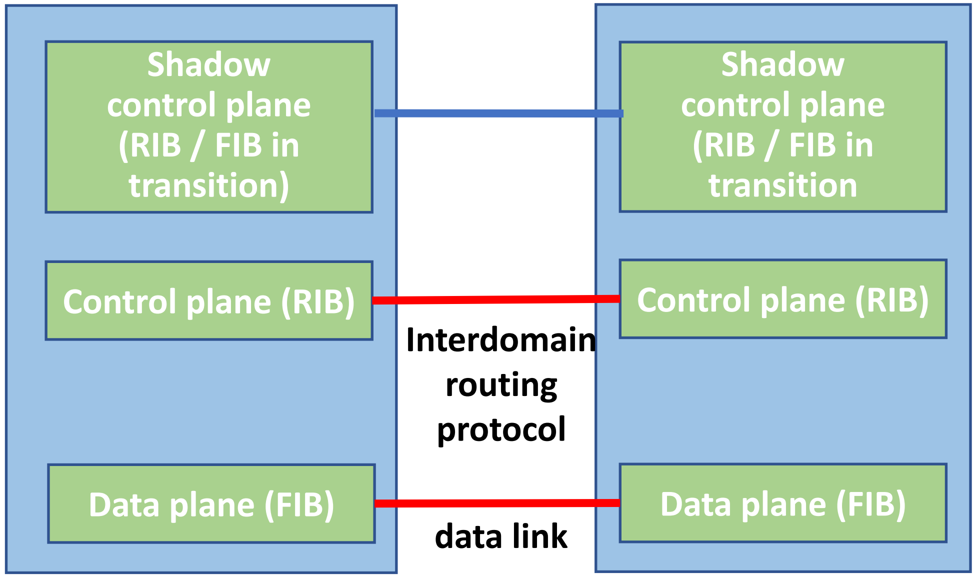
Path selections at SDI-nets must be consistent

Select Consistent Paths

- **Issue:** when a client selects a different path-port at downstream, path ports at upstream may change, causing churns and disruption

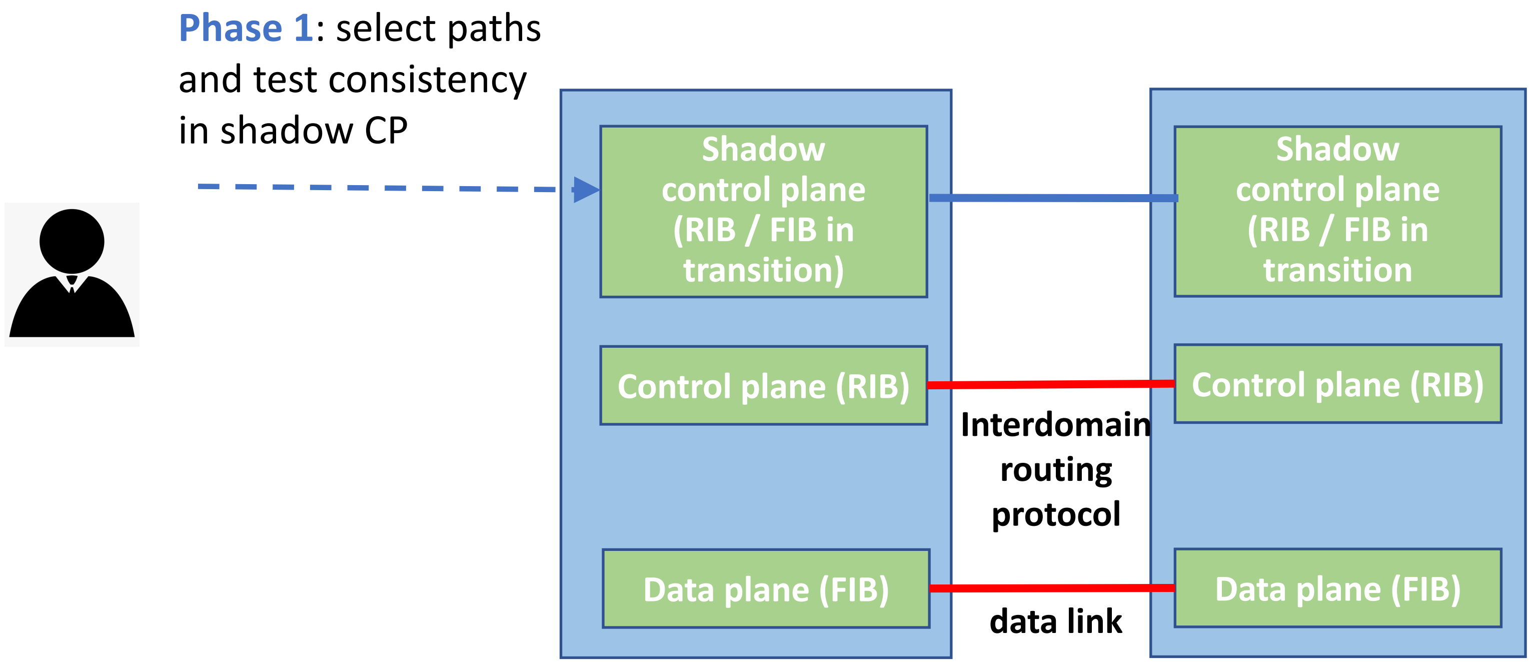
Select Consistent Paths

- **Issue:** when a client selects a different path-port at downstream, path ports at upstream may change, causing churns and disruption
- **Solution**: (1) Three-layer design of SDI-net, (2) Two-phase-commit path selection



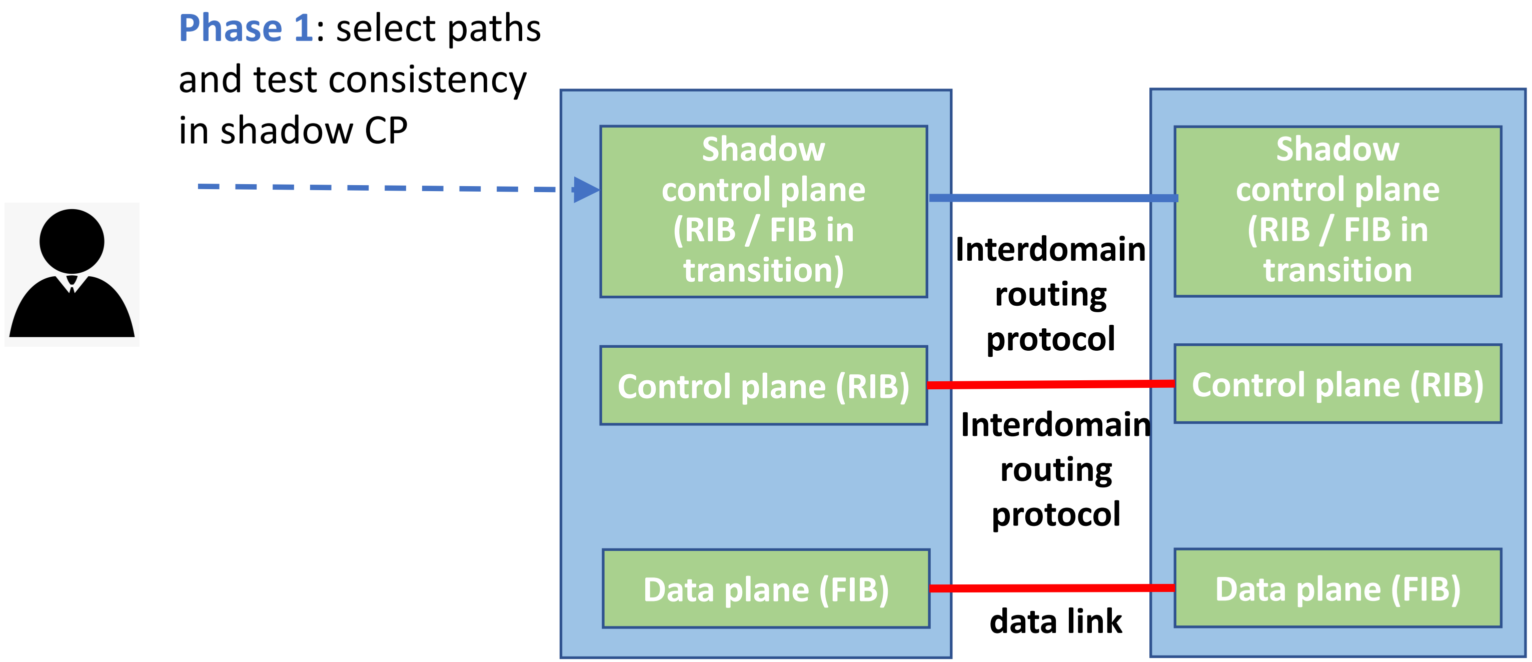
Select Consistent Paths

- **Issue:** when a client selects a different path-port at downstream, path ports at upstream may change, causing churns and disruption
- **Solution**: (1) Three-layer design of SDI-net, (2) Two-phase-commit path selection



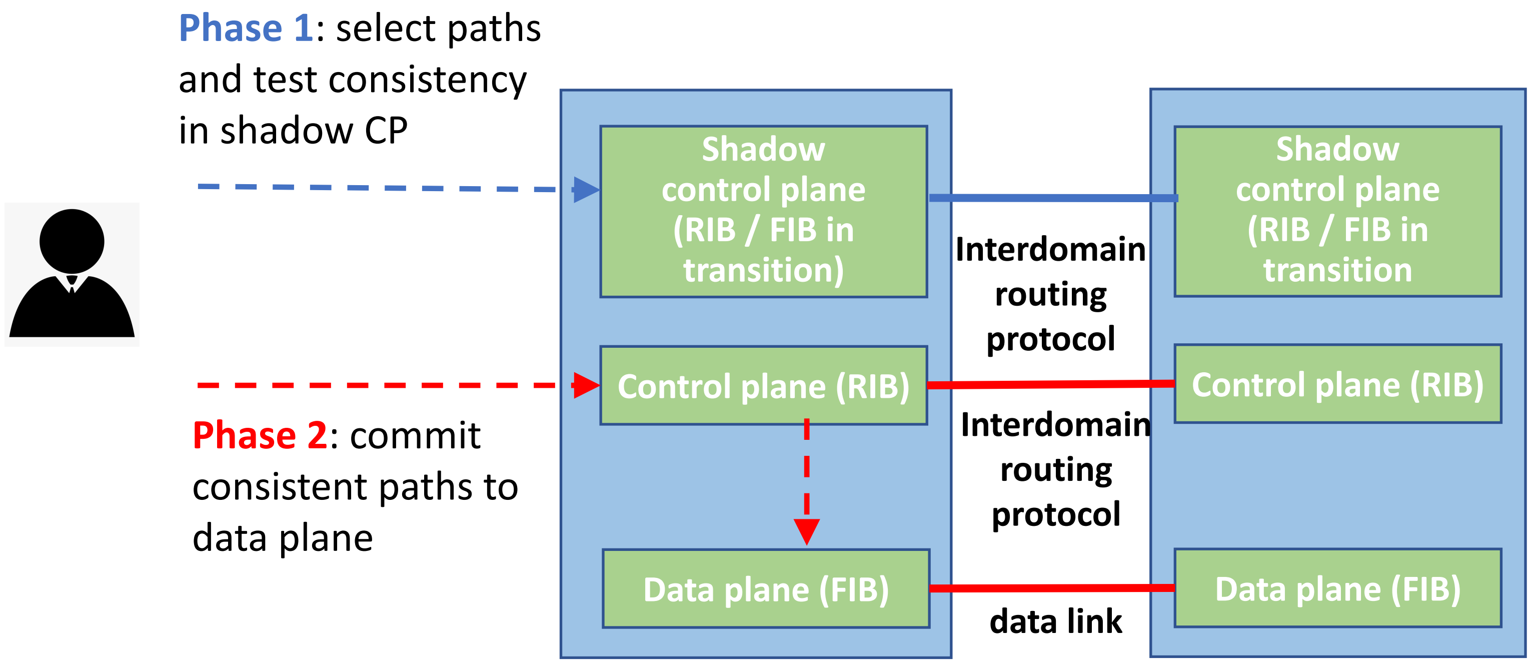
Select Consistent Paths

- **Issue:** when a client selects a different path-port at downstream, path ports at upstream may change, causing churns and disruption
- **Solution**: (1) Three-layer design of SDI-net, (2) Two-phase-commit path selection



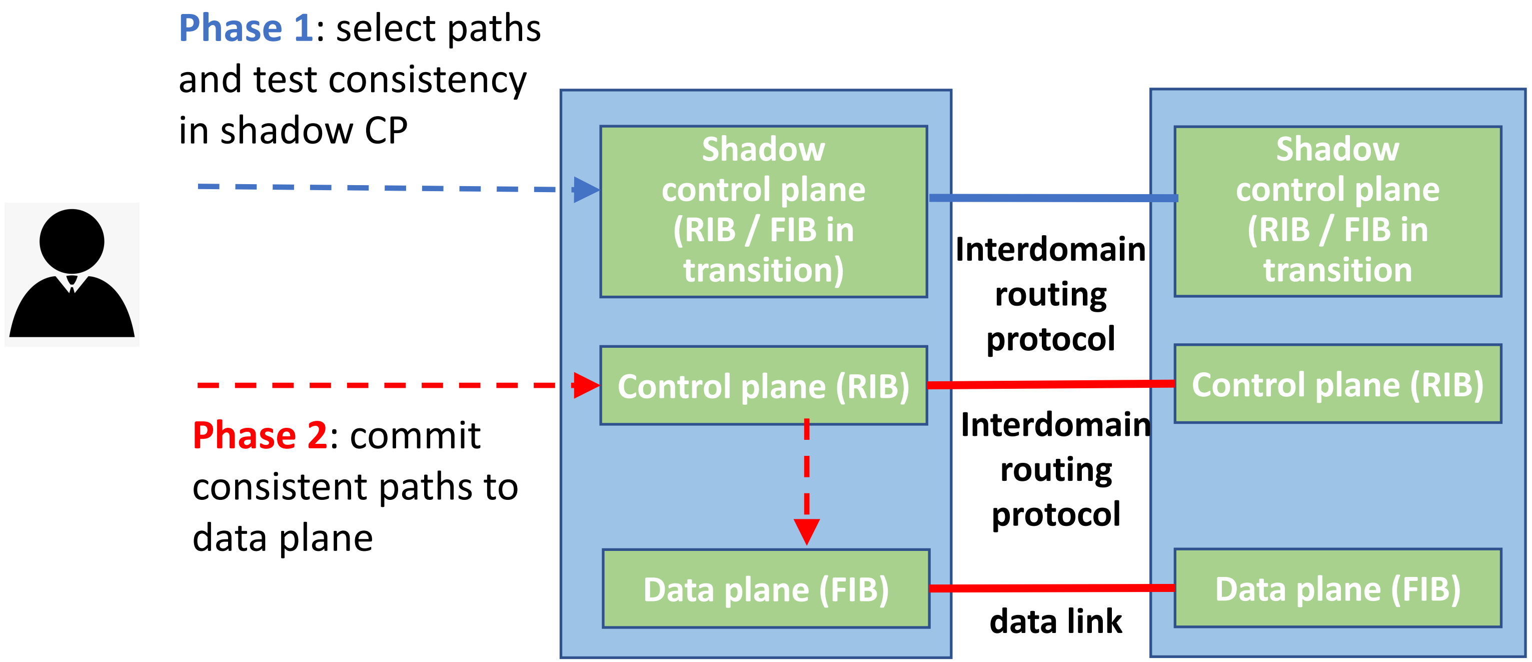
Select Consistent Paths

- **Issue:** when a client selects a different path-port at downstream, path ports at upstream may change, causing churns and disruption
- **Solution**: (1) Three-layer design of SDI-net, (2) Two-phase-commit path selection



Select Consistent Paths

- **Issue:** when a client selects a different path-port at downstream, path ports at upstream may change, causing churns and disruption
- **Solution**: (1) Three-layer design of SDI-net, (2) Two-phase-commit path selection



Outline

- Introduction
- SDI network control model
- Client SDI control optimization

Client SDI Control Optimization Problem

- **Formulation**:
  - A client connects to K SDI-nets to select consistent paths of M source-destination flows
  - Client objective defined by a utility function $c()$ over M flows

Client SDI Control Optimization Problem

- **Formulation**:
  - A client connects to K SDI-nets to select consistent paths of M source-destination flows
  - Client objective defined by a utility function $c()$ over M flows

$$\text{maximize} \quad c(\overrightarrow{r})$$

subject to,

$$\overrightarrow{r} \in R : R_1, \times \ldots, \times R_M,$$

$$r_i = \{r_i^j\}_j, \text{where } j = 1, \ldots, K.$$

Client SDI Control Optimization Problem

- **Formulation**:
  - A client connects to K SDI-nets to select consistent paths of M source-destination flows
  - Client objective defined by a utility function $c()$ over M flows

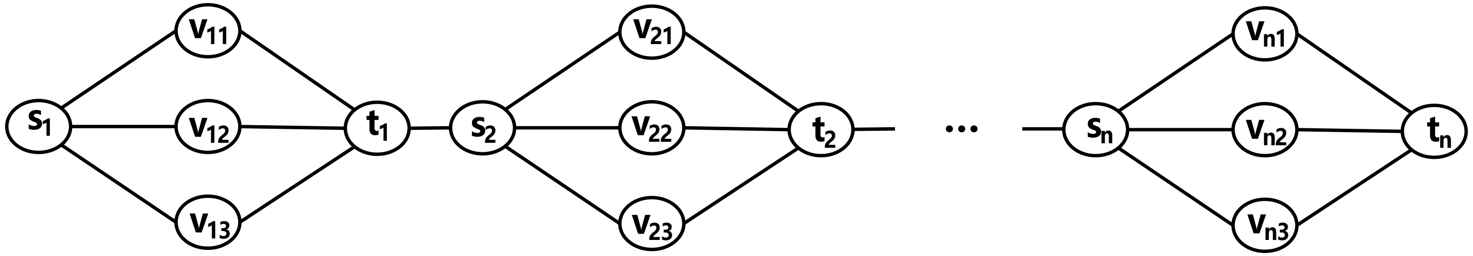
$$\text{maximize} \quad c(\overrightarrow{r})$$

$$\text{subject to,}$$

$$\overrightarrow{r} \in R : R_1, \times \ldots, \times R_M,$$

$$r_i = \{r_i^j\}_j, \text{where } j = 1, \ldots, K.$$

- **Complexity**: This problem is NP-hard via a reduction from 3-SAT problem.



A Blackbox Optimization Reformulation: Lift Path Consistency from Constraint to Objective Function

$$\text{maximize} \quad c(\overrightarrow{r})e(\overrightarrow{r})$$

Binary variable to indicate path consistency

subject to,

$$\overrightarrow{r} \in R^a : R_1^a \times \ldots, \times R_m^a.$$

$$r_i = \{r_i^j\}_j, \text{where } j = 1, \ldots, k.$$

A Blackbox Optimization Reformulation: Lift Path Consistency from Constraint to Objective Function

$$\text{maximize} \quad c(\overrightarrow{r})e(\overrightarrow{r})$$

Binary variable to indicate path consistency

subject to,

$$\overrightarrow{r} \in R^a : R^a_1 \times \ldots, \times R^a_m.$$

$$r_i = \{r_i^j\}_j, \text{where } j = 1, \ldots, k.$$

- **Basic idea**: uses the prior belief to direct the search, and uses the posterior to update the belief

A Blackbox Optimization Reformulation: Lift Path Consistency from Constraint to Objective Function

$$\text{maximize} \quad c(\overrightarrow{r})e(\overrightarrow{r})$$

Binary variable to indicate path consistency

subject to,

$$\overrightarrow{r} \in R^a : R^a_1 \times \ldots, \times R^a_m.$$

$$r_i = \{r_i^j\}_j, \text{where } j = 1, \ldots, k.$$

- **Basic idea**: uses the prior belief to direct the search, and uses the posterior to update the belief
- **Improving search efficiency**: (1) one path inconsistency can prune a large search space; (2) one consistent path can avoid many repeated tests in future search

Outline

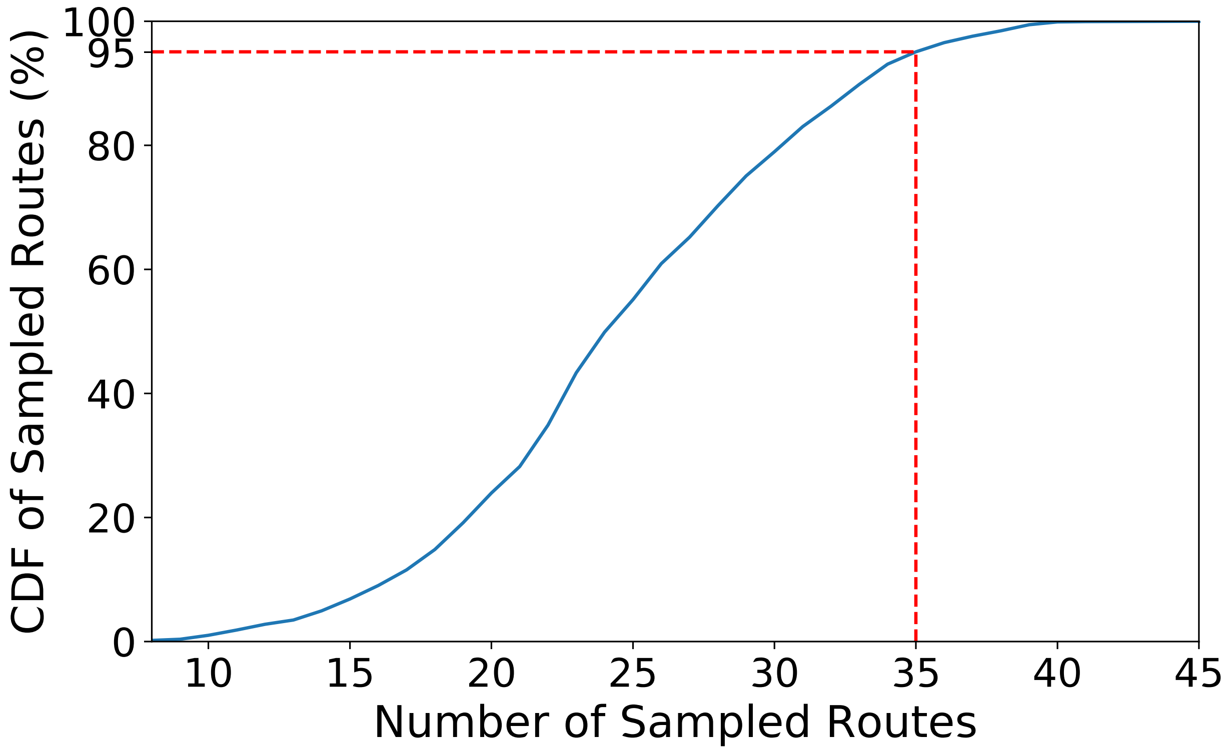
- Introduction
- SDI network control model
- Client SDI control optimization
- Evaluation

Performance Evaluation: Settings

- **Topology**: CAIDA Internet topology dataset with 63361 ASes and 320978 AS-level links.
- **AS export policies**: (1) C/P relationship, (2) blacklist ASes, (3) forbidden segments.
- **Client objective**: find shortest AS path for top 2000 AS-pairs in terms of traffic volume, based on CAIDA Internet traffic dataset

Results: Efficacy and Efficiency of SDI Control

- In all experiments, the SDI optimization algorithm finds the optimal policy-compliant shortest AS path
- In 95% cases, it finds the optimal solution by sampling no more than 35 paths.



Outline

- Introduction
- SDI network control model
- Client SDI control optimization
- Evaluation
- Operational Implication: Privacy Study

Can BGP Policies Be Inferred from Exposed RIBs and Selected Route?

- **Perception**: BGP is usually good at hiding policies, and BGP looking glass / ALTO servers are deployed
- **Preliminary finding**: BGP selection policy can be inferred by solving a classification problem
- **Simulation setting**: 3-20 neighbor ASes, next-hop-based local preference assignment, standard route selection procedure (i.e., RFC 4271), 200-20k (RIB, selected route) samples per dataset
- **Result**: When the # of neighbor ASes is small (i.e., <=8), 160 samples in a feed-forward neural network provides a minimal of 95% accuracy

Conclusion and Future Work

- Propose the simple, novel software-defined internetworking (SDI) model, extending intradomain SDN to generic interdomain SDN
- Design an efficient optimization algorithm to solve the client SDI control optimization problem
- Demonstrate the feasibility, benefits and potential privacy concern of SDI via evaluation results

Future work

- Extend from coarse-grained (i.e., destination IP based) SDI to fine-grained (i.e., TCP/IP 5-tuple) SDI
- Accurate BGP policy inference with few-shot learning