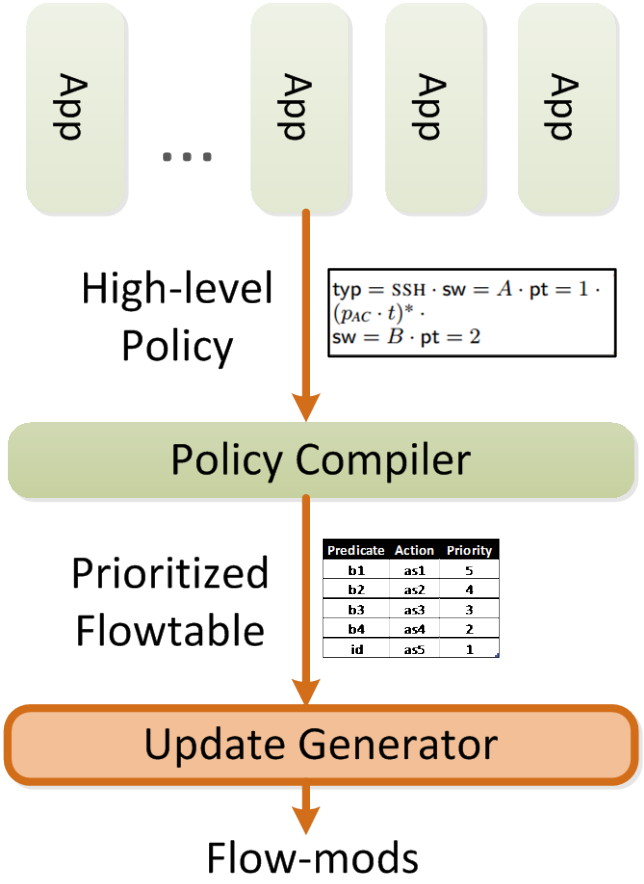
Xitao Wen, Chunxiao Diao, Xun Zhao, Yan Chen, Li Erran Li, Bo Yang, Kai Bu

Northwestern University, Tsinghua University, Bell Labs Alcatel-Lucent, Zhejiang University

Compiling Minimum Incremental Update for Modular SDN Languages

Motivation

- Flowtable update bottleneck
  - 10s to 100s of rule edits per second
  - Full refresh of 5K entries takes minutes
- **Goal**: minimizing update size to speed up flowtable update
  - Only update the "diff"



Problem Statement

- Update rules whose content or priority changes
  - 3 rule adds + 2 priority updates
- Priority updates contribute over 90% in average!

***Question:* How can we minimize priority updates?**

Diff with Reassigned Priority

- **Idea:** Modify priorities assigned by compiler
- **Challenges:**

Constraint 1: New priority assignment MUST observe rule dependency

- *Solution:* Minimum dependency construction

Constraint 2: New priority values MUST be integers within [0, 65535]

- *Solution:* Priority gap maintenance

Pattern	Priority
<1, 2>	3
<*, 2>	2
<*, *>	1

Old

Pattern	Priority
<1, 2>	~~5~~ -> 3
<2, *>	~~4~~ -> 2.5
<1, *>	~~3~~ -> 2
<*, 2>	~~3~~ -> 2
<3, *>	~~2~~ -> 1.5
<*, *>	1

New

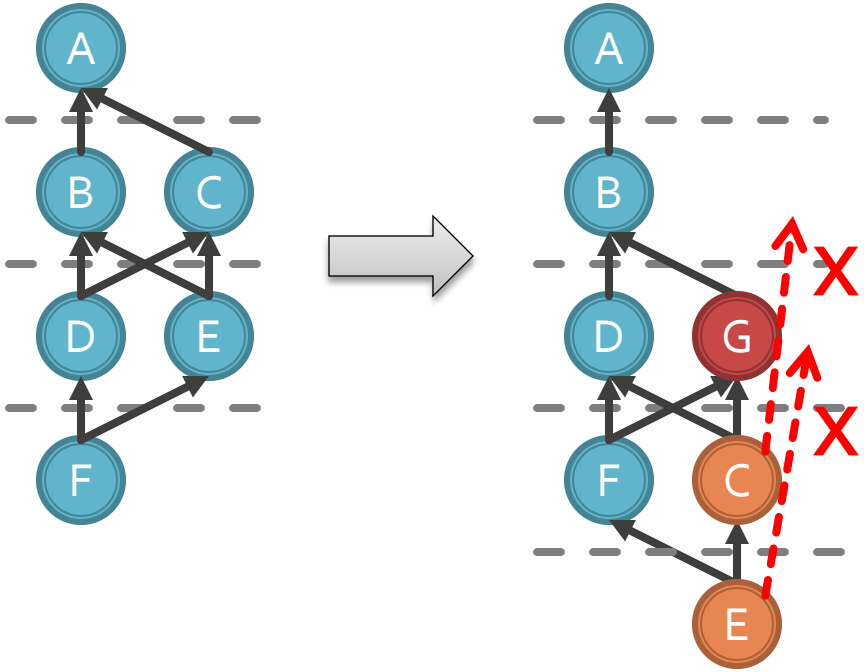
3 rule edits!

Dependency Constraint

- **Constraint 1**: rule dependency
- Dependency inferred from priority value is problematic

	Pattern	Priority
A	<1, 2, *>	5
B	<*, 2, 3>	4
C	<1, *, 4>	4
D	<1, *, 3>	3
E	<*, *, 4>	3
F	<*, *, 3>	2

Old



	Pattern	Priority
A	<1, 2, *>	5
B	<*, 2, 3>	4
G	<*, 2, 4>	3
C	<1, *, 4>	2
D	<1, *, 3>	3
E	<*, *, 4>	1
F	<*, *, 3>	2

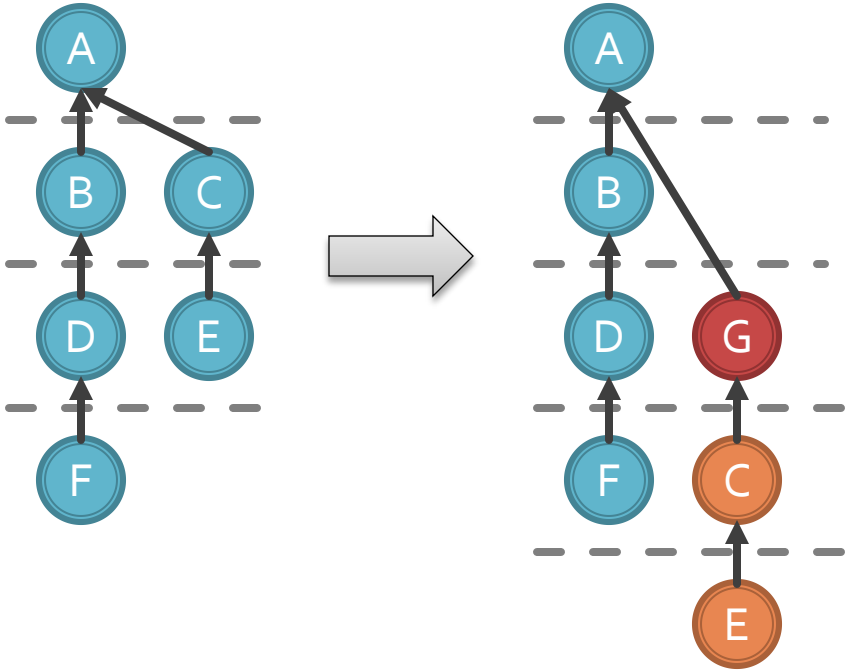
New

Dependency Constraint

- Minimum dependency can be inferred from rule patterns

	Pattern	Priority
A	<1, 2, *>	5
B	<*, 2, 3>	4
C	<1, *, 4>	4
D	<1, *, 3>	3
E	<*, *, 4>	3
F	<*, *, 3>	2

Old



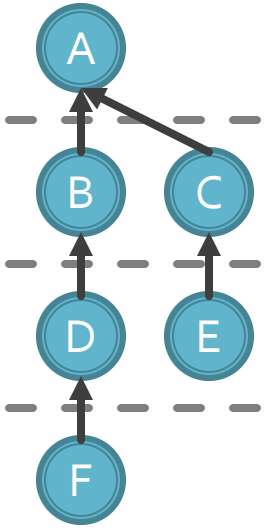
	Pattern	Priority
A	<1, 2, *>	5
B	<*, 2, 3>	4
G	<*, 2, 4>	3
C	<1, *, 4>	**2**
D	<1, *, 3>	3
E	<*, *, 4>	**1**
F	<*, *, 3>	2

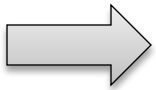
New

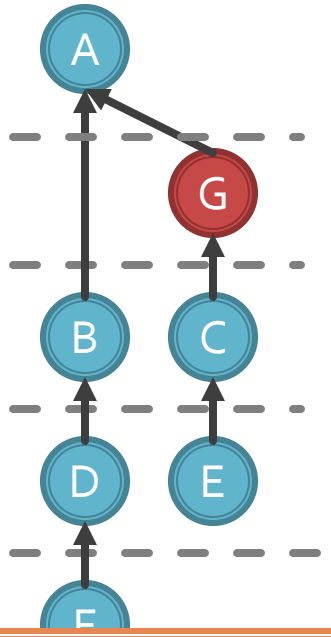
Dependency Constraint

- With minimum dependency graph, one can always generate minimum-size flowtable update if priority value is continuous

	Pattern	Priority
A	<1, 2, *>	5
B	<*, 2, 3>	4
C	<1, *, 4>	4
D	<1, *, 3>	3
E	<*, *, 4>	3
F	<*, *, 3>	2







	Pattern	Priority
A	<1, 2, *>	5
B	<*, 2, 3>	4
G	<*, 2, 4>	~~3~~-> 4.5
C	<1, *, 4>	~~2~~ -> 4
D	<1, *, 3>	3
E	<*, *, 4>	~~1~~ -> 3
F	<*, *, 3>	2

***Take-away:* Minimum dependency helps eliminate priority updates**

How to obtain minimum dependency

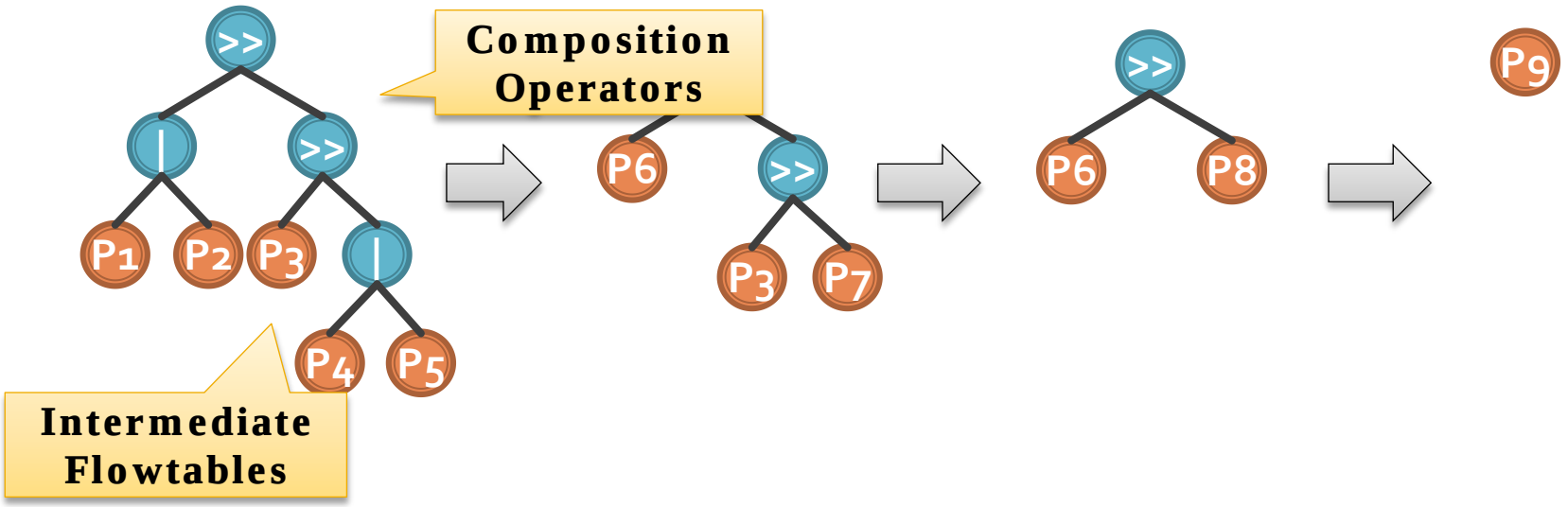
- Restored from prioritized flowtable after compilation
  - Incurs complicated header space computation

- Constructed along with compilation
  - Rule dependency can be recursively inferred from policy composition process
  - Incurs little additional overhead over compilation

* Find the algorithm description and the complexity analysis in our technical report at http://goo.gl/gBLBrm

Incremental Dependency Construction

- Recursive construction of minimum dependency
  - Extend intermediate flowtables with dependency graph
  - Keep track of dependency during compilation
    - Parallel composition
    - Sequential composition



Incremental Dependency Construction

- Infer dependency for parallel composition
  1. Graph cross-product
  2. Intersection tests
  3. Special treatment for compiler-specific data structure
- Sequential composition is similar except for the actions of the first operands

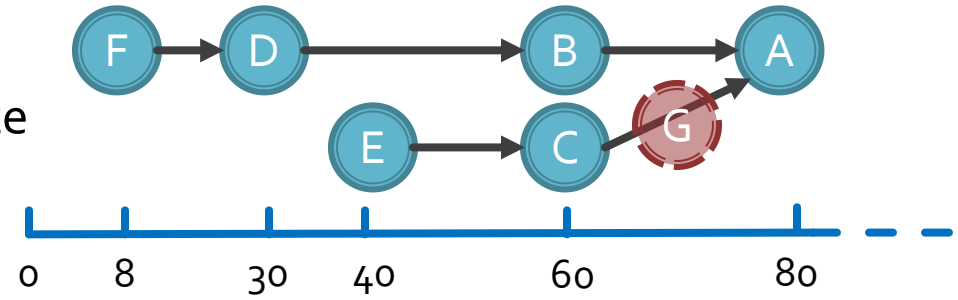
Maintaining Priority Value Distribution

- **Constraint 2**: discrete priority values
  - Integers ranging [0-65535] for OpenFlow
  - If new rule is inserted between adjacent priority values, we have to shift existing rules to make room for them
- Problem Statement
  - Assign priority values for priority levels
  - Objective: minimize the estimation of priority shifts
- Online strategy
  - Undetermined future policy update sequence

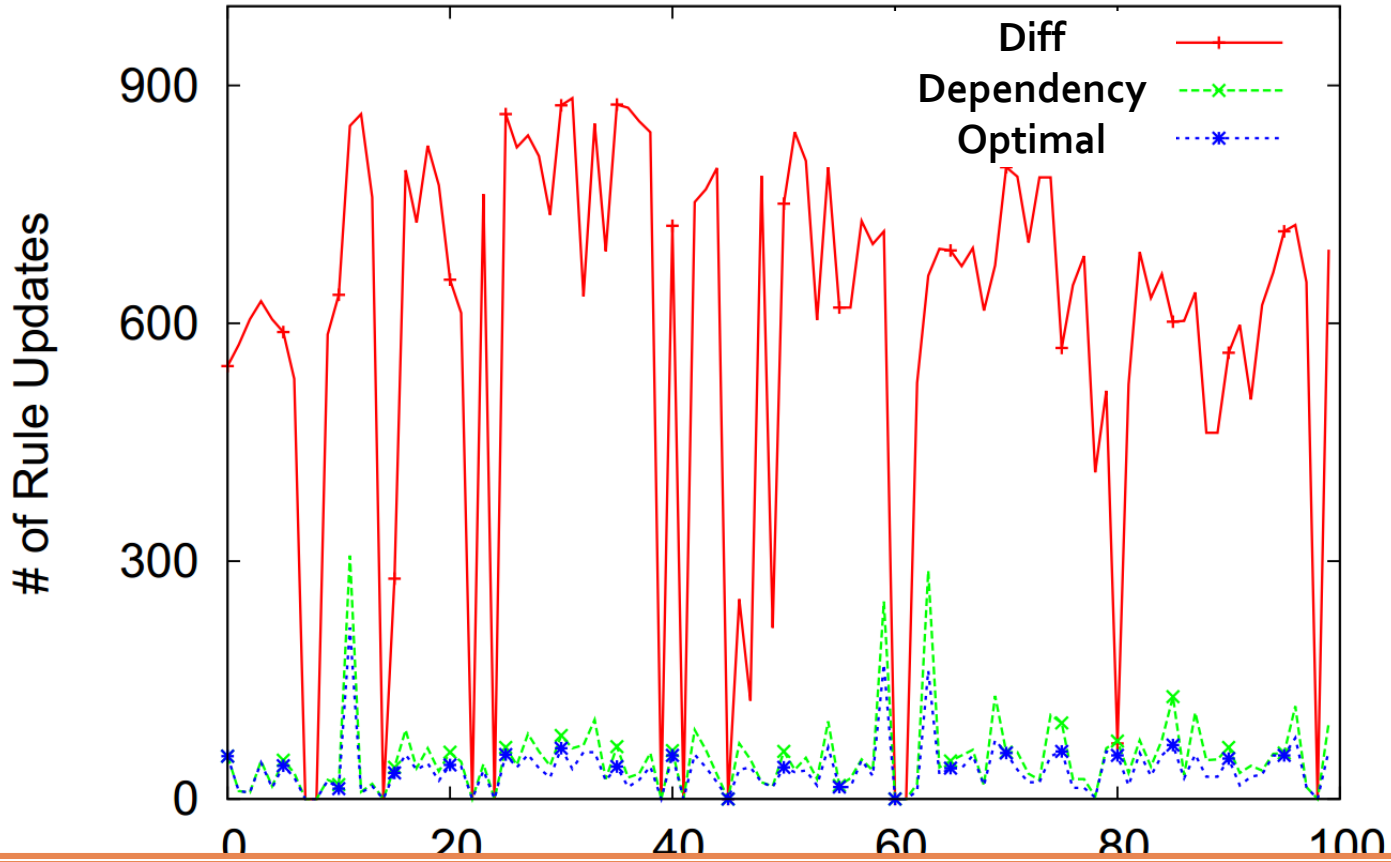
Pattern	Priority
<1, 2>	~~5~~ -> 3
<2, *>	~~4~~ -> 2.5
<1, *>	~~3~~ -> 2
<*, 2>	~~3~~ -> 2
<3, *>	~~2~~ -> 1.5
<*, *>	1

K-factor strategy

- Key idea: proactively maintains the ratio between max priority gap and min priority gap
- Initiation
  - Distribute priority levels evenly on [0-65535]
- Invariance
  - Keep lengths of all gaps between [1/*k*, *k*] * mean length, where *k* is a parameter
- Cost
  - Amortized: O(1) per update



Evaluation



- ***Eliminates almost all priority updates***
- ***10x smaller on average compared to diff***

Conclusion

- Minimum incremental update framework comprising of
  - Minimum update generation with dependency
  - K-factor strategy for priority gaps maintenance
- Future work
  - Dependency construction algorithms generic to policy languages
  - Lower bound of priority gap maintenance

Xitao Wen
xitaowen2015@u.northwestern.edu

Question?