JIVE
Performance Driven Abstraction and Optimization for SDN
ONS 2014 Research Track

Aggelos Lazaris (USC), Daniel Tahara (Yale),
Xin Huang (CYAN), Li Erran Li (Bell Labs), Andreas Voellmy (Yale),
Y. Richard Yang (Yale), Minlan Yu (USC)
Motivation

● SDN objectives
  ○ simpler and easier programming of networks
  ○ reduce controller-switch dependency
  ➢ rely on a **single switch model**

● [OpenFlow](https://www.opennetworking.org/) cannot resolve the diversity of switch implementations, capabilities, and behaviors
  ○ e.g. TCAM size, TCAM management
Motivation

Insertion of the same sequence of rules results in low throughput in the first switch, and rule rejection in the second switch.
**Motivation**

Insertion of the *same sequence of rules* results in **low throughput** in the first switch, and **high throughput** in the second switch.
Switch Diversity

● Diversity in flow **tables types** and **table sizes**
  ○ software tables, hardware tables (TCAM), or both
  ○ various TCAM sizes
    ■ 369 - ~10K rules
    ■ table size might vary depending on the matching fields
      ● L2/L3, L2+L3
Switch Diversity

- Diversity in **flow installation** behaviors, and **data plane delay** using different flow tables
  - 3 (or 2) tier delay observed
    - **fast path**
      - packets matching rules in the TCAM
    - **slow path (in some switches)**
      - packets matching rules in the software table
    - **control path**
      - packets matching no rules
Switch Diversity

- Diversity in controller-switch channel performance
  - delay to update rules << delay to install new rules
  - delay to install rules in descending priority order >>
    delay to install rules in ascending priority order
    ■ up to 6 times smaller
JIVE

- Objectives
  - reveal switch capabilities
  - introduce abstractions to unify switch diversity
  - API

- Design
  - infer JIVE patterns
  - optimization, scheduling
JIVE Patterns

- JIVE pattern is a sequence of flow_mod commands, and a corresponding data traffic pattern
  - infer patterns
    - infer flow table size
    - infer cache algorithms
JIVE Abstractions

- Unify switch diversity
  - abstract 2-layer architecture
    - different flow table sizes
    - different installation behaviors
- expose JIVE functionality to the application through an API
  - e.g. setup latency, bandwidth
JIVE Optimization

- Scheduling & Routing
  - compute and set up a path for each request
  - expression rewriting
    - rewrite the flow rules such that we minimize the installation time
      - e.g. ascending priority, topological ordering
    - potentially introduce additional paths where a dummy flow entry is installed and later modified
JIVE Architecture
JIVE Evaluation

Flow Installation Time

Installation time of 1K Classbench rules

Up to 12X improvement
Summary

- **JIVE**
  - abstractions
    - unify switch diversity
    - API
  - optimization using expression rewriting and scheduling

- **Future directions**
  - better understanding the features of the various hardware switches (e.g. multiple tables, etc.)
End of Presentation

Thank You!

Email: alazaris@usc.edu
Supporting Slides