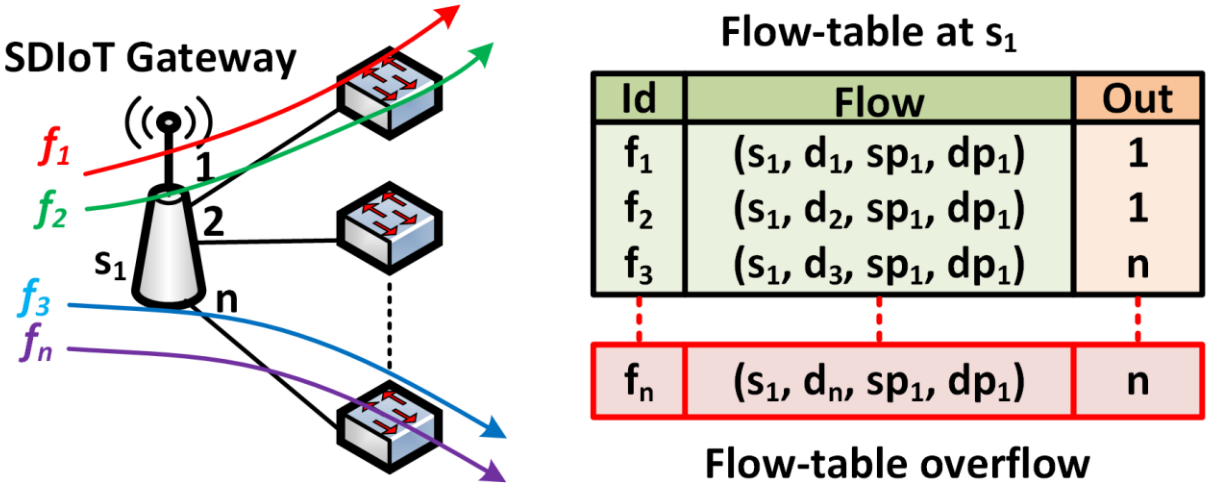
QoS-Aware Adaptive Flow-Rule Aggregation in Software-Defined IoT

N. Saha, S. Misra and S. Bera
Department of Computer Science and Engineering,
Indian Institute of Technology, Kharagpur, India

IEEE GLOBECOM 2018, Abu Dhabi, UAE

Problem Statement

- SDN utilizes the OpenFlow protocol for rule-based data-plane operations.
- Flow-rules are in the form of match-action pairs, with each rule capable of matching on multiple fields such as ingress port, vlan id, ethernet, and tcp header fields.
- TCAM memory in OpenFlow switches is limited.
- Fine-grained QoS forwarding uses exact-match rules.

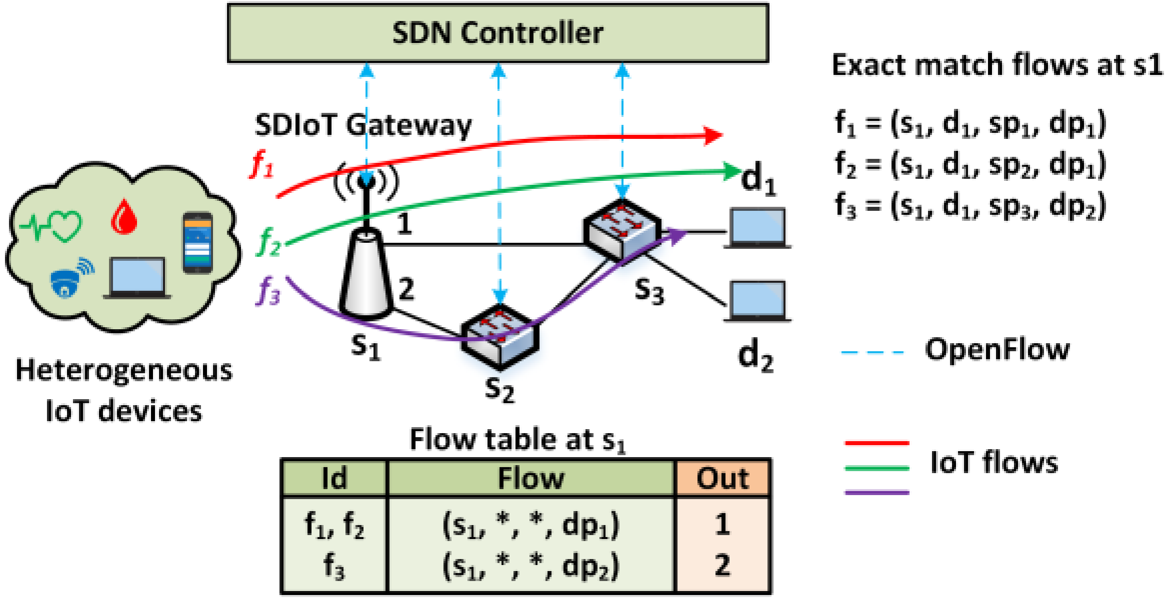


Flow-table overflow due to exact-match rules

There is a need to address the flow-table overflow problem

N. Saha, S. Misra and S. Bera, Indian Institute of Technology, Kharagpur

Problem Statement (cont.)



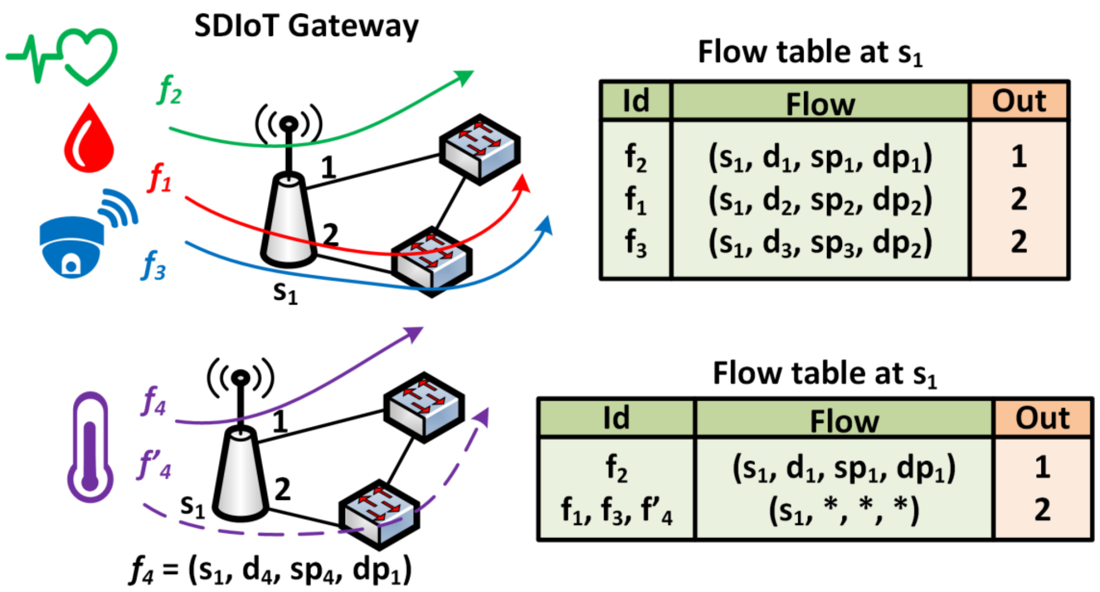
System Architecture

- Heterogeneous IoT connected to SDN-enabled backbone by SDIoT gateways.
- Flow-rule $r_j = \langle M_j, A_j, C_j \rangle$
  - M_j -> match fields
  - A_j -> action set
  - C_j -> counters
- Flow table at switch s_i is given as $R_i = \{r_j^i \mid 1 \leq j \leq R^{max}\}$

- IoT flows require application specific QoS treatment.
- Fine grained QoS forwarding using exact-match rules lead to rule-overflow.
- Aggregating the flow-rules using a combination of source and destination port i.e., (s1, ∗, ∗, dp1) is capable of correctly forwarding the IoT flows under consideration.

N. Saha, S. Misra and S. Bera, Indian Institute of Technology, Kharagpur

Problem Statement (cont.)

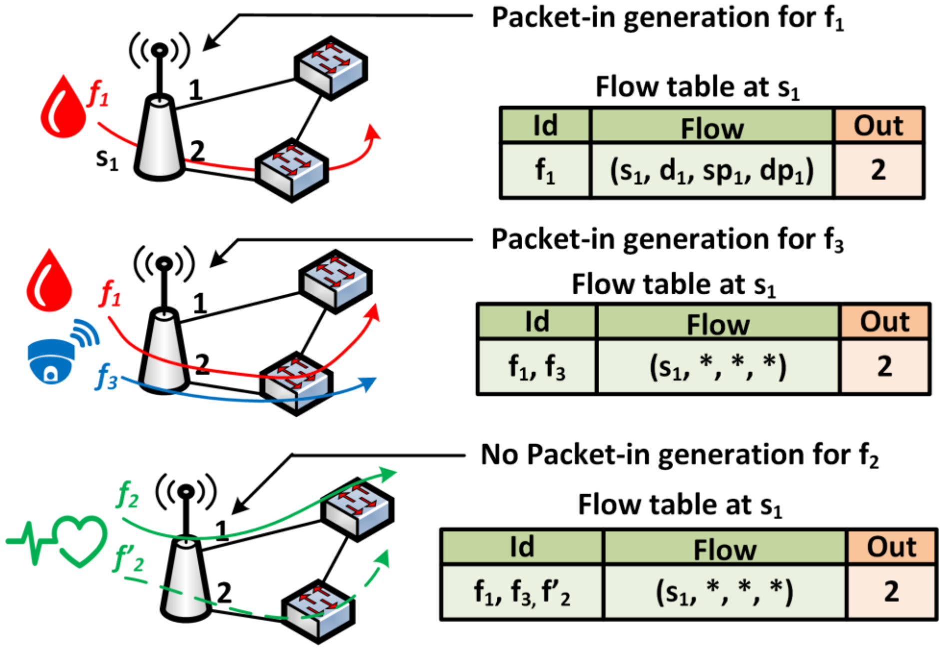


Flow table at s_1

Id	Flow	Out
f_2	(s_1, d_1, sp_1, dp_1)	1
f_1	(s_1, d_2, sp_2, dp_2)	2
f_3	(s_1, d_3, sp_3, dp_2)	2

Flow table at s_1

Id	Flow	Out
f_2	(s_1, d_1, sp_1, dp_1)	1
f_1, f_3, f'_4	$(s_1, *, *, *)$	2



Flow table at s_1

Id	Flow	Out
f_1	(s_1, d_1, sp_1, dp_1)	2

Flow table at s_1

Id	Flow	Out
f_1, f_3	$(s_1, *, *, *)$	2

Flow table at s_1

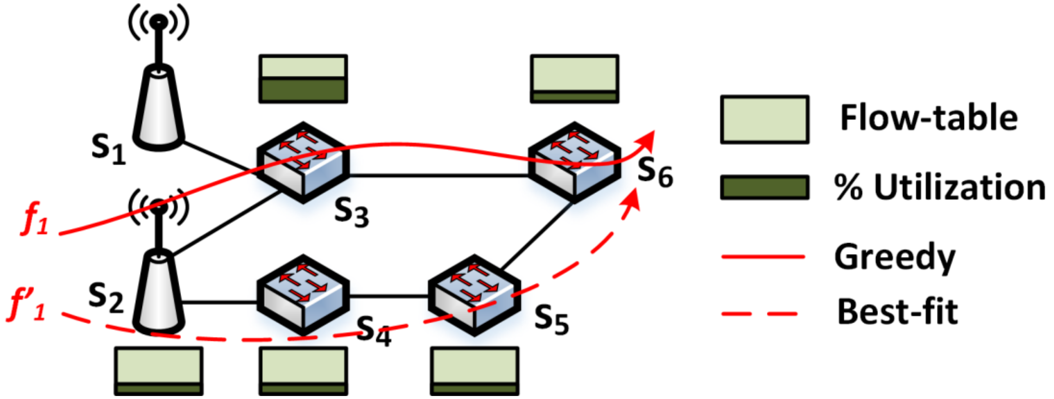
Id	Flow	Out
f_1, f_3, f'_2	$(s_1, *, *, *)$	2

Rifai *et al.* (IEEE GLOBECOM 2015). At s1, the correct output action for flows from s1 with dst port dp1 is out port 1. However, due to the (s1, ∗, ∗, ∗) rule, f4 is forwarded incorrectly out port 2.

Kosugiyama *et al*. (IEEE ICC2017). Packet-in messages are generated for flows f1 and f3 due to table-miss. However, flow f2 matches the aggregated flow rule (s1, ∗, ∗, ∗) and is forwarded incorrectly out port 2, before generation of packet-in message.

Adaptive Flow-Rule Aggregation

- Need to choose from **multiple candidate paths.**
- Flow-table overflow at **bottleneck switch** invalidates all paths through that switch.



Given a set of paths, choose the path P with minimum cost $\delta(P)$. The cost of choosing a path P is given as

$$\delta(P) = \sum_{s_i} \left(\alpha\lambda + \beta \max_i \left(\frac{|R_i|}{R^{max}} \right) \right)$$

where λ represents the cost of inserting a new flow-rule and α, β are normalizing constants.

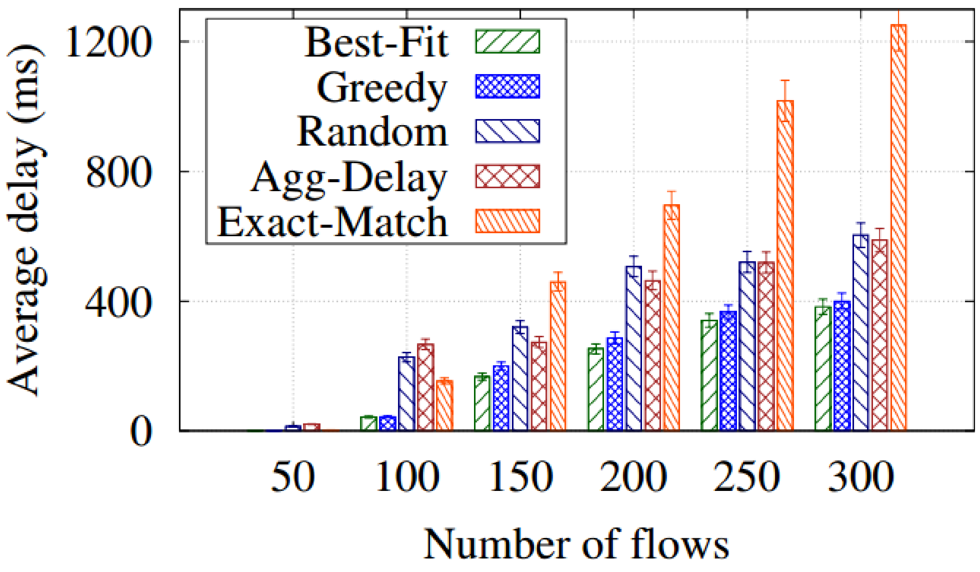
The greedy approach chooses path f1 with three new flow-rule insertions at s2, s3 and s6. The Best-fit heuristic takes into account the bottleneck switch, s3, and chooses path f10 with four new flow-rule insertions at s2, s4, s5 and s6.

N. Saha, S. Misra and S. Bera, Indian Institute of Technology, Kharagpur

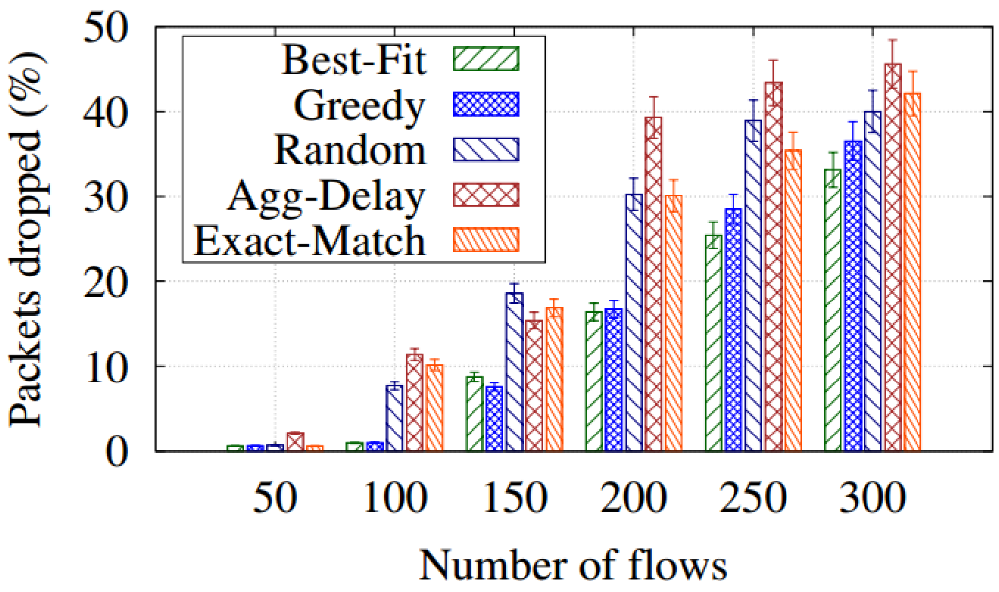
Solution Approach

for each $s_i \in \mathcal{S}$ **do**
    initialize empty dictionary, d_i
for each flow $f_k \in \mathcal{F}$ **do**
    get set of QoS paths, $\mathcal{P} = \{P_k \mid 1 \leq k \leq |\mathcal{F}|\}$
    get the least cost path P_k using the **Best-fit** heuristic
    for each switch $s_i \in P_k$ **do**
        AGGREGATE exact-match rule r_k
procedure AGGREGATE
    extract key λ_k from r_k using user-defined match-fields
    if $\lambda_k \in keys(d_i)$ **then** ▷ Rule exists, aggregate
        append value $d_i[\lambda_k] \leftarrow r_k$
        create modified rule r'_k using key λ_k, while wildcarding rest of the match fields
        update rule r'_k in flow-table of s_i
    else if $\lambda_k \notin keys(d_i)$ **then** ▷ Rule does not exist
        insert key-value pair $d_i[\lambda_k] \leftarrow r_k$
        place r_k in flow-table of s_i

Performance Evaluation



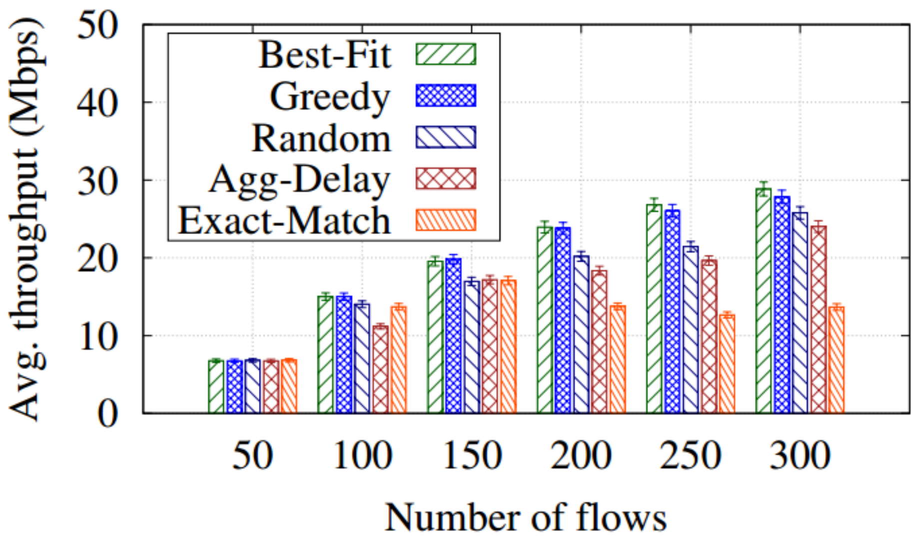
Average end-to-end delay



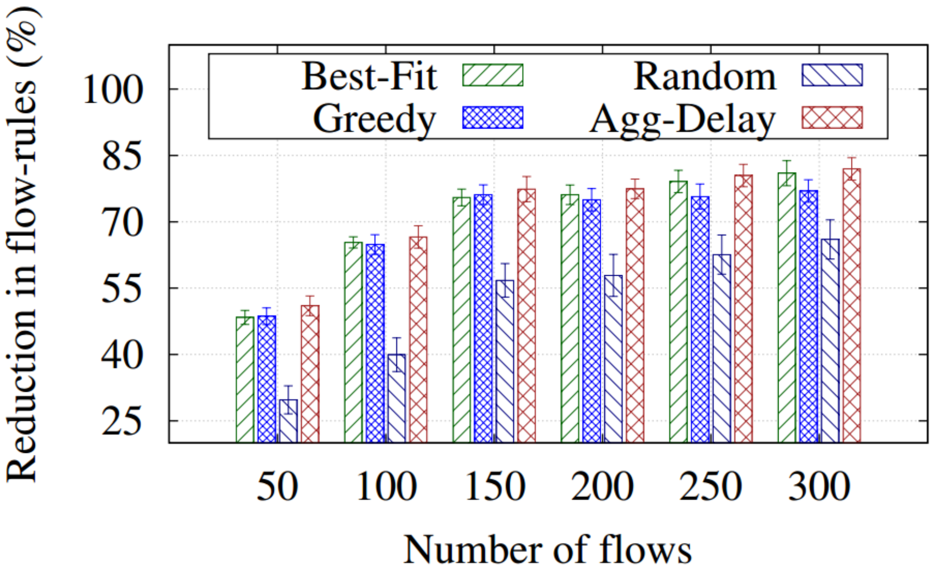
Average packet-loss

- With 300 flows in the network, the proposed scheme reduces the average delay by 35% and 70% and packet loss by 10% and 12% compared to Agg-Delay and Exact-match, respectively.
- Exact-match suffers due to the effect of flow-setup delay for every flow.
- Agg-Delay incurs more loss due to wrong forwarding decisions.

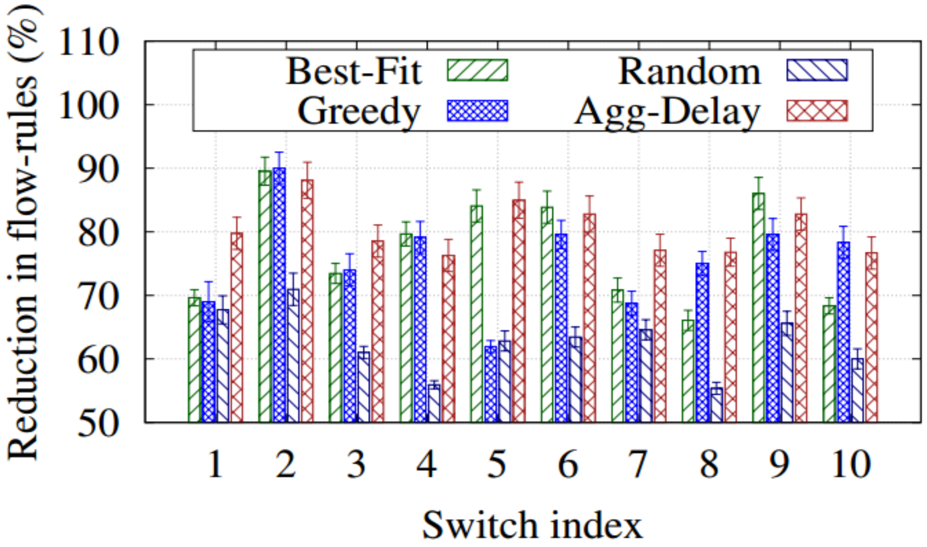
Performance Evaluation (cont.)



Average throughput



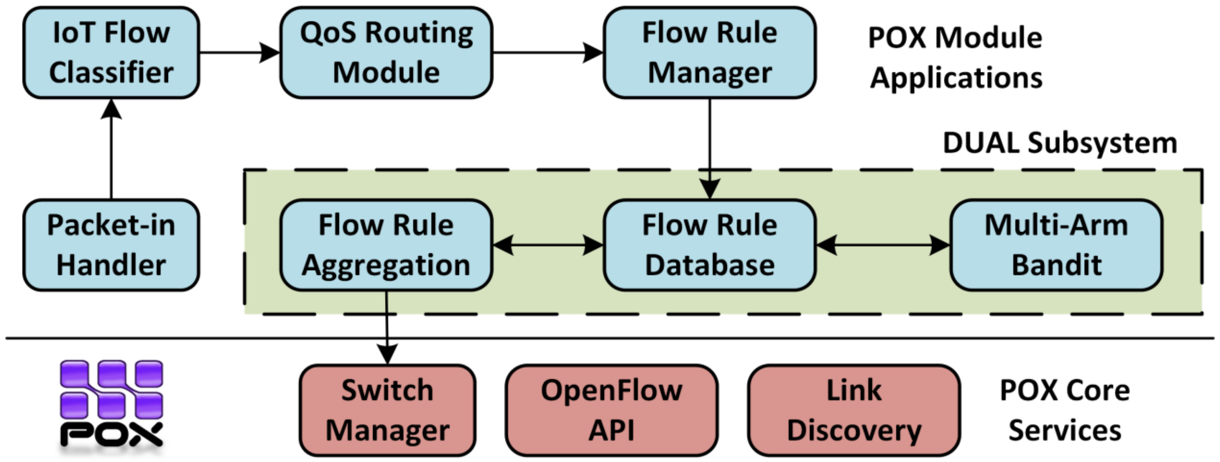
Reduction in flow rules



- The proposed scheme incurs 20%and 110% increase in throughput compared to Agg-Delay and Exact-match, respectively.
- The Best-fit heuristic leads to a more uniform distribution of flow-rules across the network.

N. Saha, S. Misra and S. Bera, Indian Institute of Technology, Kharagpur

Current Work in Progress



The proposed scheme consists of three components:

- Key-based aggregation scheme capable of fast flow-rule aggregation.
- Multi-arm bandit (MAB)-based scheme for selecting the best key.
- Best-fit heuristic to maximize the total number of flow-rules that can be placed in the network.

- OpenFlow 1.5 specification supports upto 44 header fields.
- If more number of match-fields are considered, QoS violations will decrease at the cost of increase in the flow-table size.
- Which one of the *k*-combinations will lead to optimal trade-off between number of flow-rules number of QoS-violated flows?

THANK YOU

N. Saha, S. Misra and S. Bera, Indian Institute of Technology, Kharagpur