Reinforcement learning based load balancing for data center networks

Jiyoon Lim

Supervisor: Prof. James Won-Ki Hong

Dept. of CSE, DPNM Lab., POSTECH, Korea

limjiyoon@postech.ac.kr

2020. 12. 22

Table of Contents

Introduction

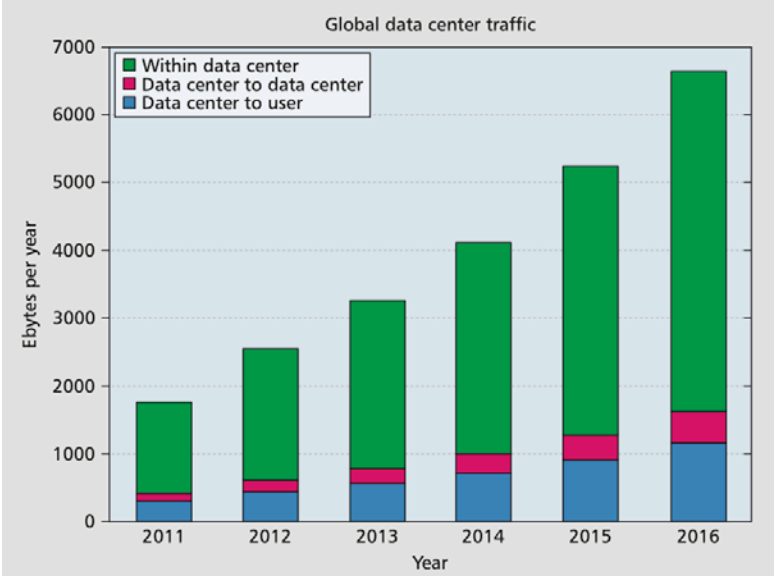
Introduction

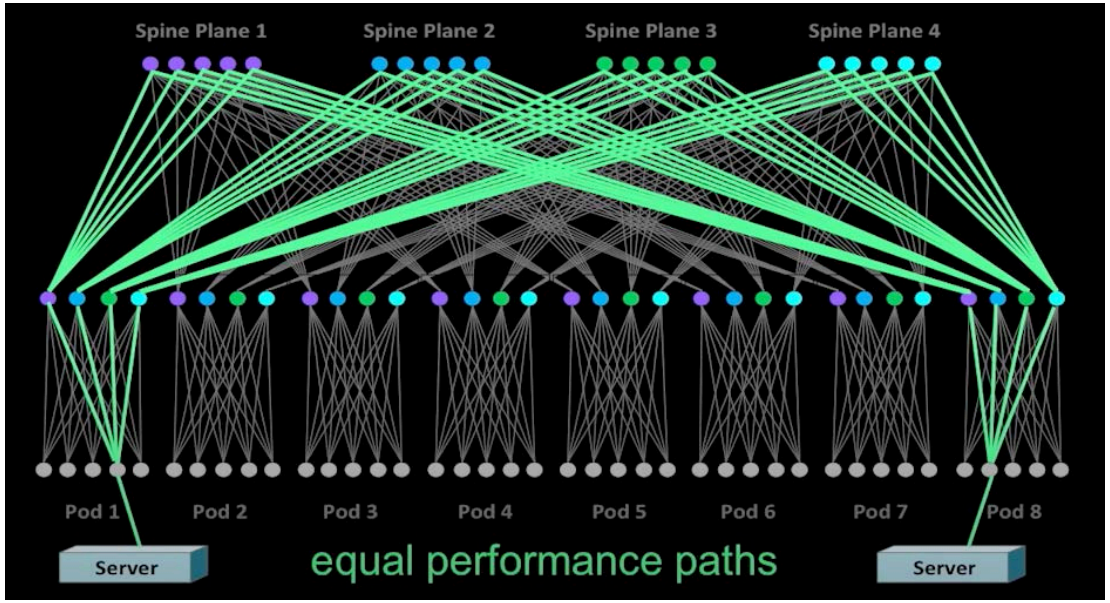
❖ Data center network (DCN)

- Large facilities that compute, storage, and network working in concert
- To support increasing cloud services, many DCNs have been deployed

❖ Characteristics of DCNs

- Increased server-to-server communication
  - East-west traffic : 70 ~ 80%
- Provide Large bisection bandwidth
  - Use Clos topology with high path diversity
  - Load balancing is a key to achieving large bisection bandwidth
- Provide resilience service





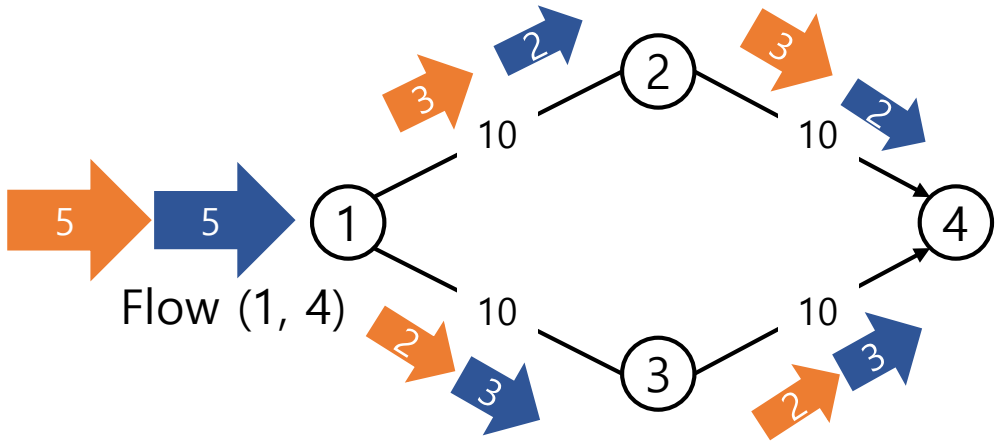
https://www.youtube.com/watch?v=mLEawo6OzFM

Introduction

❖ Load balancing problem

- Modeled as a Minimum cost Multicommodity flow problem (MCMF)
- Goal : Find the flow assignments to minimize Network utilization
  - Network utilization : $\sum_{(u,v)\in E} U(u,v)^2$
  - Link utilizations : $U(u,v) = \frac{\sum_{i=1}^{K} f_i(u,v)\cdot d_i}{c(u,v)}$
    - $f_i(u,v)$: pass ratio of i^{th} flow at link (u,v)
    - d_i : Traffic size of i^{th} flow
    - $c(u,v)$: capacity of link (u,v)
- NP-hard problem
  - If fractional flows are allowed: polynomial time

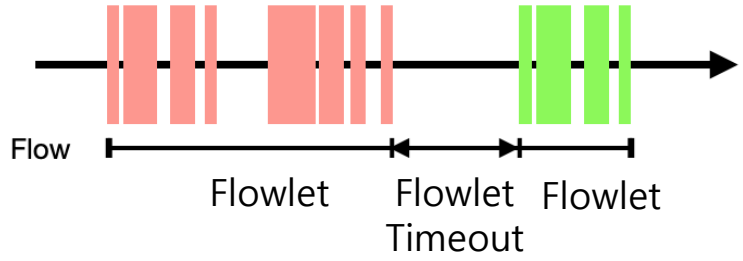
Network utilization : 1

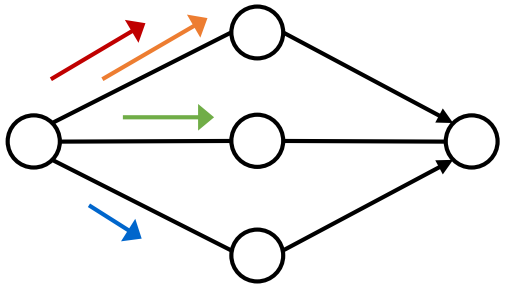


Minimum cost multi commodity flow problem

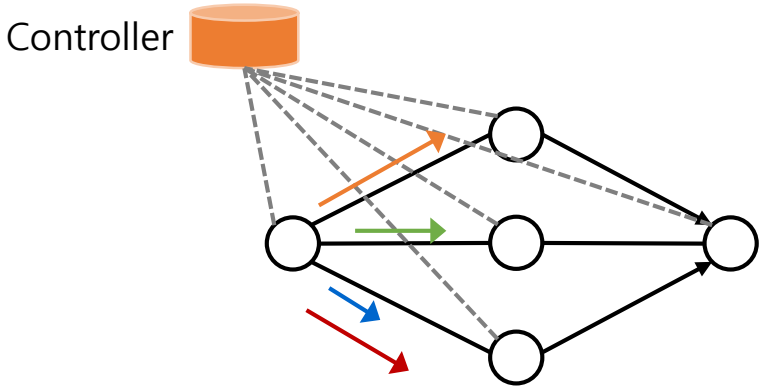
❖ Load balancing in DCN

- Equal-cost Multipath(ECMP)
  - Evenly distributes flows to each path
  - Hash collision problem : Possible to cause congestion if routes two elephant flows to same path
- Using network information with centralized controller
  - Hedera (NSDI 10) : Shows 2× higher bisection bandwidth than ECMP
- Using smaller scheduling granularity
  - Flare : Splits flows into flowlets at each switch
    - Flowlet : burst of packets that is separated in time from other bursts

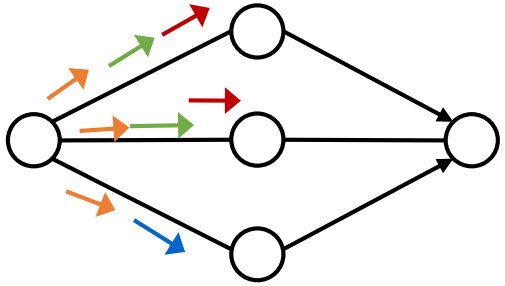




ECMP

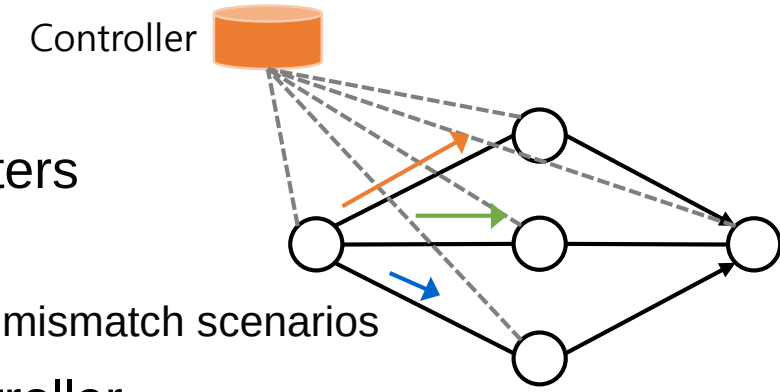


Using network information



Flowlet unit routing

Introduction



❖ Problem statement

- Significant performance degradation occurs with improper parameters
  - Needs to set parameter for traffic pattern and topology automatically
  - PIAS load balancing algorithm shows 38.46% performance degradation under mismatch scenarios
- Decide late for most of flows due to communication delay with controller
  - Cannot handle mice flows (few kb) which occupy majority of flows in DCNs
  - e.g) 60ms delay → 7.5MB size traffic flows in 1Gbps → apply routing decision to other flows

❖ Research Goals

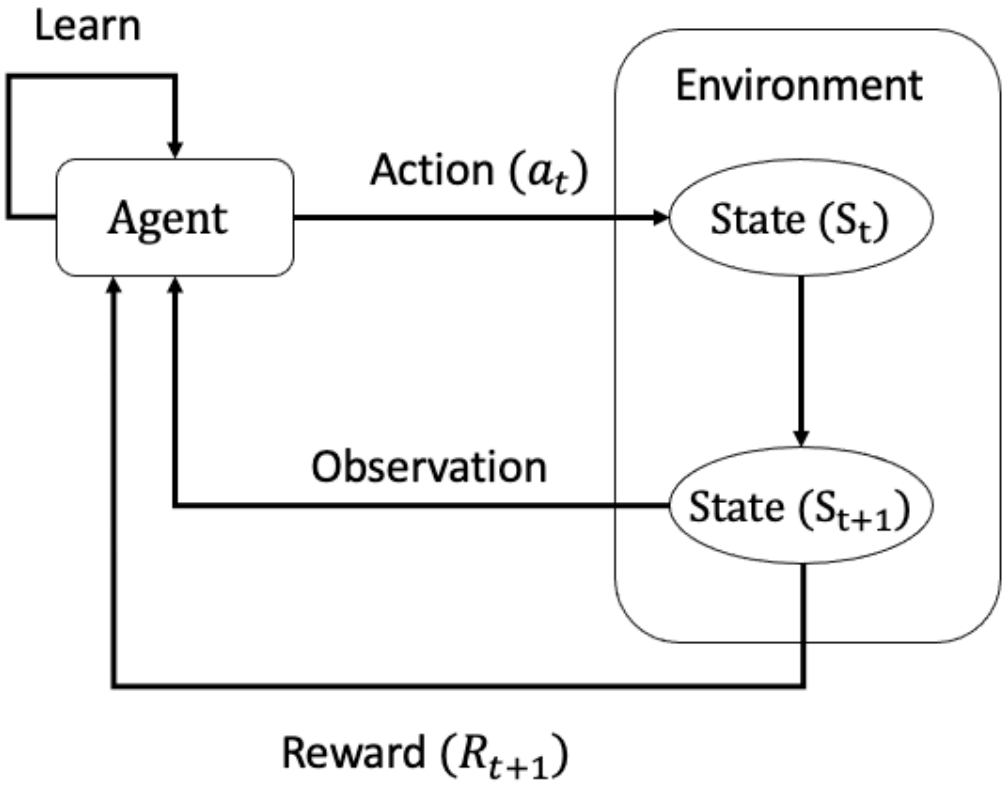
- Finds flow assignments for a traffic pattern and topology that minimize network utilization
  - Learns traffic split ratios of egress ports for each node with reinforcement learning
- Finds an alternative approach to handle mice and elephant flows both
  - Split flows into flowlets and Make routing decision with multiple flows in advance

Background & Related work

Background

❖ Reinforcement learning

- Machine learning paradigm that learns how agent take actions in an environment
  - Agent : Learner
  - Action : agent's behavior that effect to environment
    - Policy : function that maps states to action
  - Environment : agent cannot control
  - Observation : limited view of environment
  - Reward : feedback on actions
    - Q(s,a) : expected cumulative reward when performs action a at state s
- Goal : maximize cumulative reward
- Strength
  - Consequence decision making
- Weakness
  - Partially observable
  - Large or unknown delay in the observation / reward



Related work

❖ DCN load balancing algorithm

- Conga (Sigcomm 14)
  - Store network information into each router
  - Each router makes decision based on network information
  - 20% lower flow completion time than ECMP at heavy load (50~80% load)
- LetFlow (NSDI 17)
  - Find optimal flowlet timeout and do ECMP
  - Shows similar performance with Conga

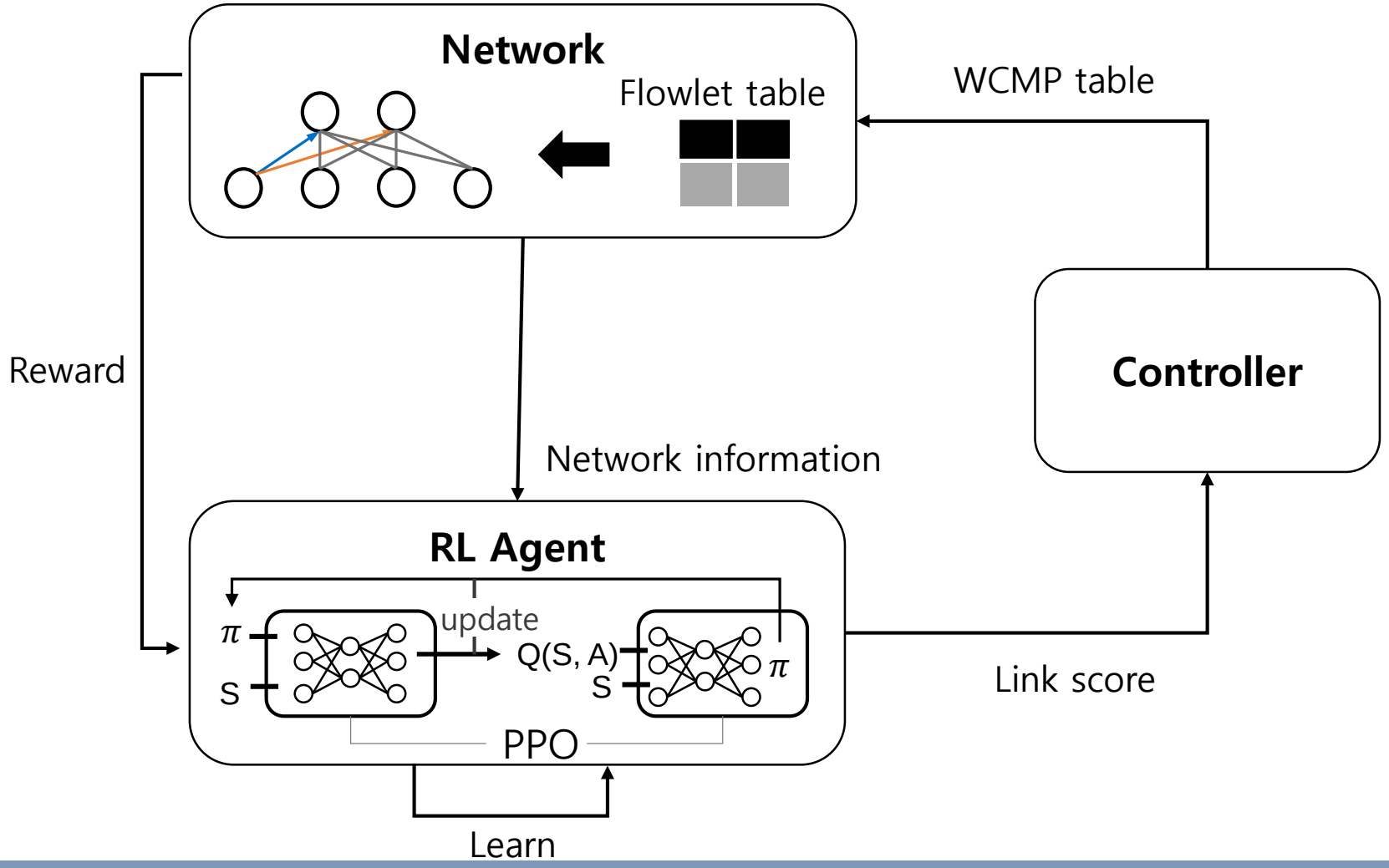
❖ Reinforcement learning based traffic optimization

- AuTo (Sigcomm 18)
  - Presents the problem of communication overhead when apply reinforcement learning in DCNs
  - Use ECMP to mice flows and reinforcement learning to elephant flows
- Learn to route with RL (NIPS 17)
  - Shows reinforcement learning algorithm yield better performance than supervised learning

Design and Implementation

❖ Reinforcement learning-based weight-cost multipathing(RWCMP)

- Weight : Traffic split ratio



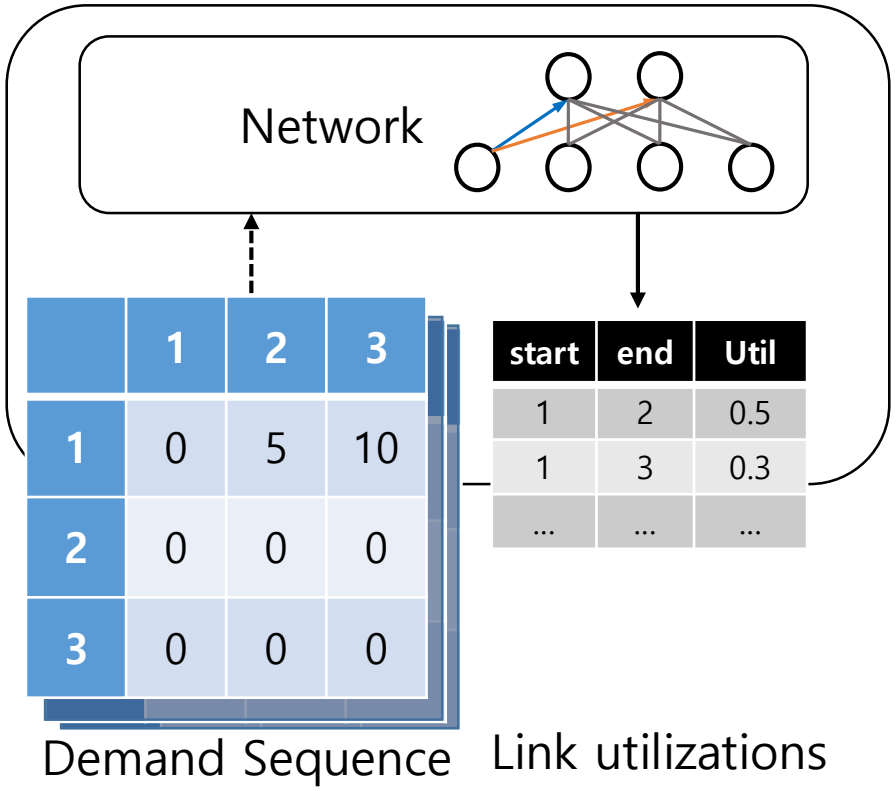
Environment

❖ Network

- Assumption on topology : Clos topology
- Clos topology
  - Each switch is connected to all switches in the neighboring stage

❖ State

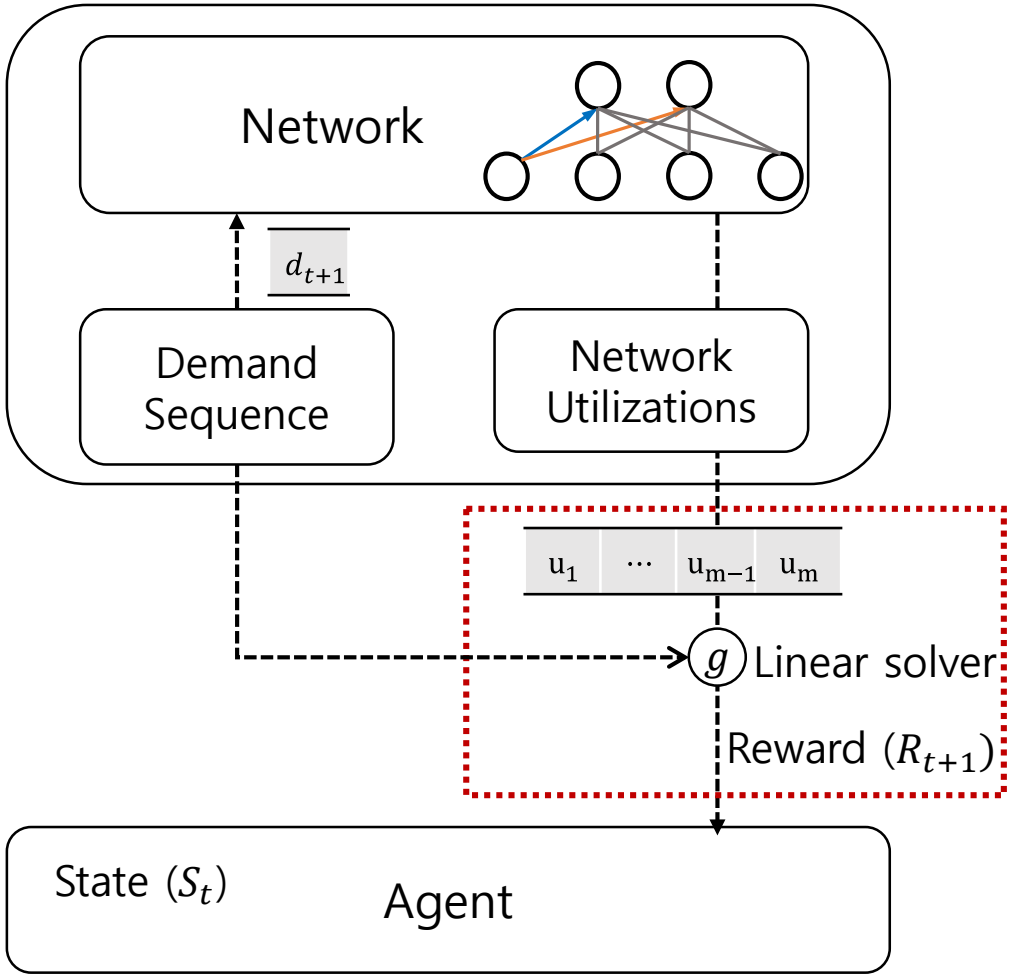
- Demand sequences
  - Sequence of traffic matrix
  - Traffic matrix : set of traffic demands for every end-host
  - Traffic demands : outgoing flow size
  - Size : # flows × $(\#\, nodes)^2$
- Link utilizations
  - Set of link utilization for every edge in a network
  - Size : # edges



Environment

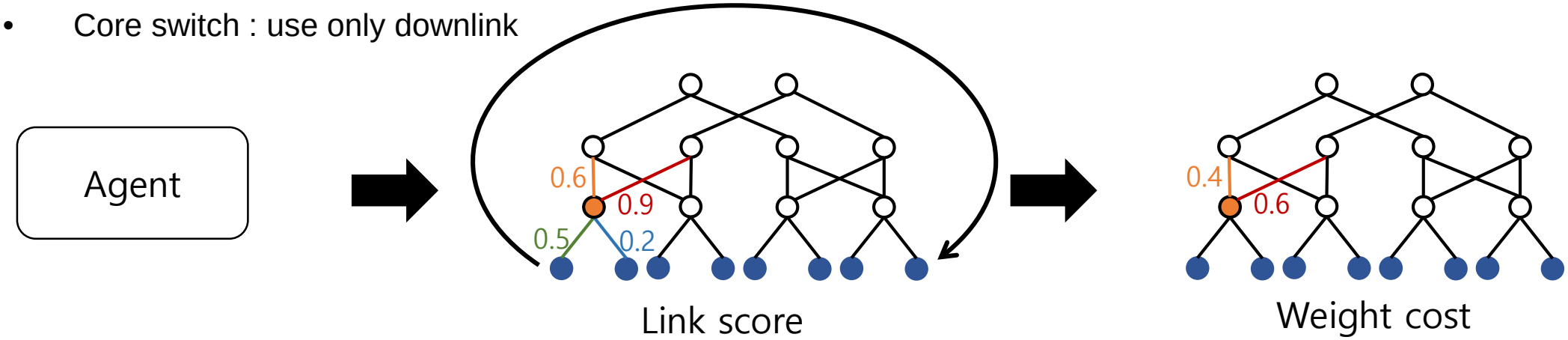
❖ Reward

- -1 × Network utilization (NU)
  - Use network utilization from MCMF problems
  - $\mathrm{NU} = \sum_{(u,v)\in E} U(u,v)^2$
- -1 × Maximum link utilization Ratio (MLU Ratio)
  - $\mathrm{MLU} = \max_{(u,v)\in E} U(u,v)$
  - Give a hint of what is an optimal value
  - $\mathrm{MLU\_RATIO} = \frac{MLU_{agent}}{MLU_{optimal}}$



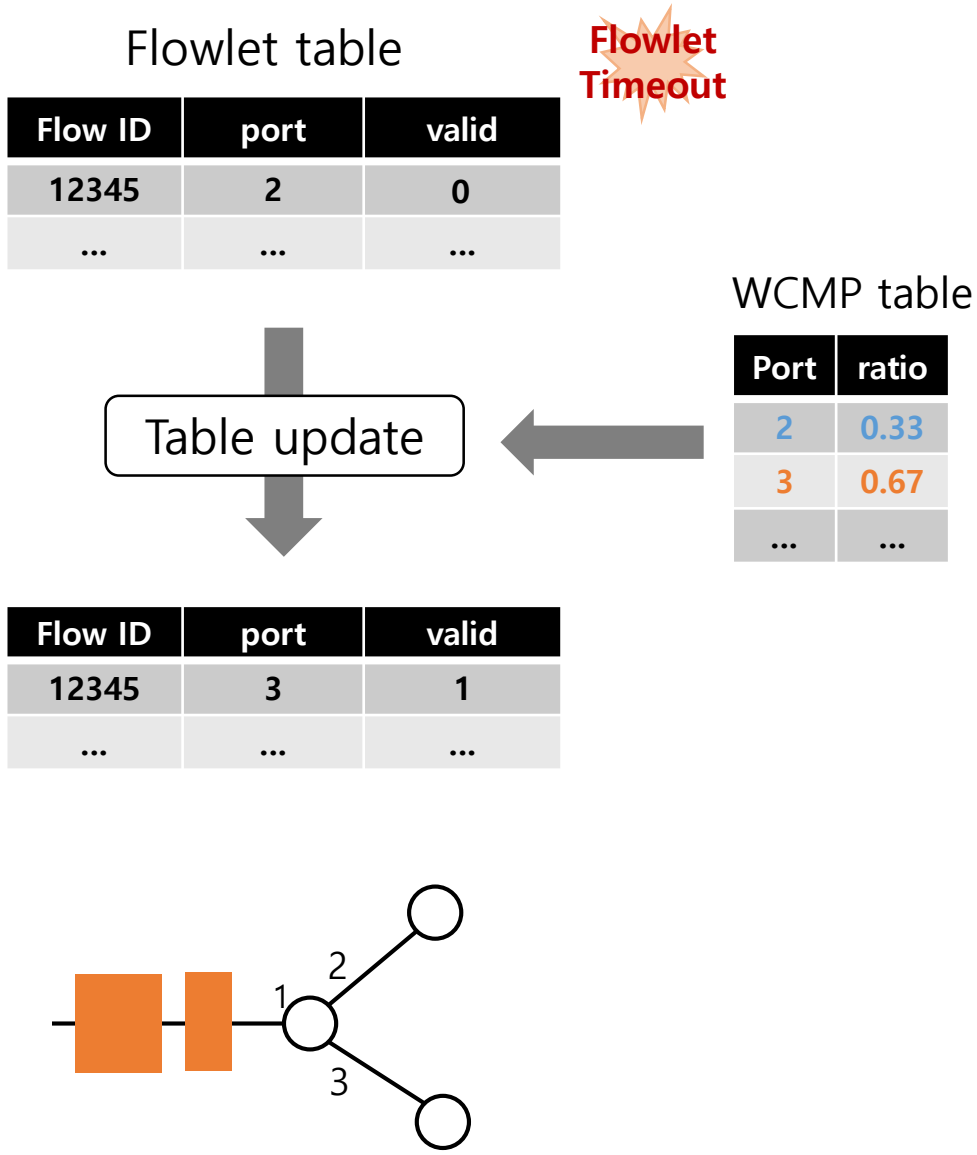
❖ Action

- Link score
  - Link score from RL model to calculate the split ratio of traffic
  - Link score ∈[0, 1]
- Weight-Cost Multipath table (WCMP Table)
  - Weight : Split ratio of egress port
    - $Weight_{ni} = \frac{\text{score}(n,i)}{\sum_{m \in \text{Node} \wedge (n,m) \in \text{Edge}} \text{score}(n,m)}$
  - Separate uplinks and downlinks to avoid loop
    - Edge / Aggregation Switch : use only uplink
    - Core switch : use only downlink



❖ Flowlet table

- Entry : (Flow ID, port, valid)
  - Flow ID : unique identifier of flow
  - Port : egress port for current flow
  - Valid : check whether ‘port’ field is valid or not
- Update port
  - When valid is 1, use current port
  - When valid is 0, update port based on WCMP table
  - Periodically unset valid bit (1 → 0)
    - Period : Flowlet timeout (500μs)

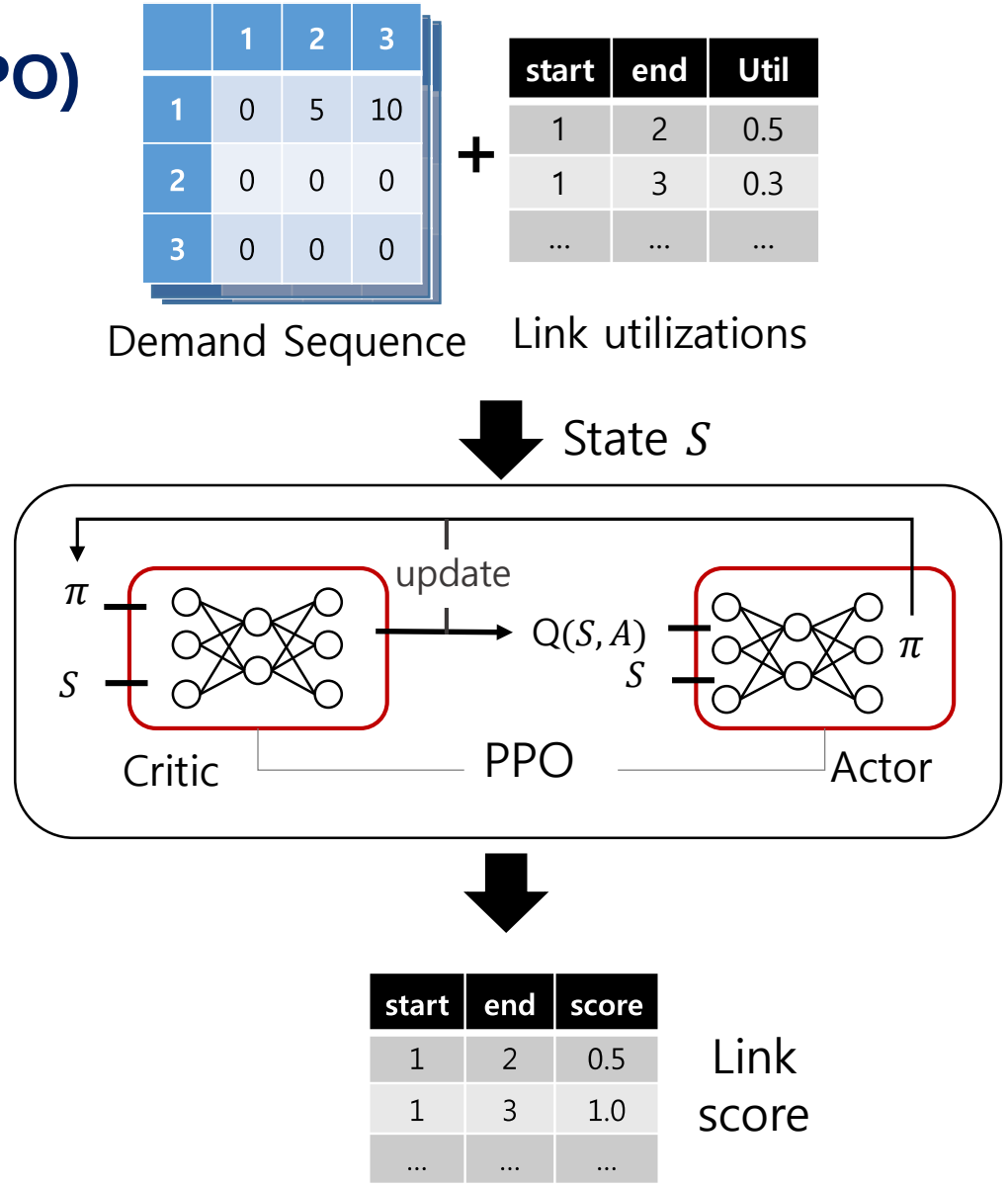


Learning algorithm

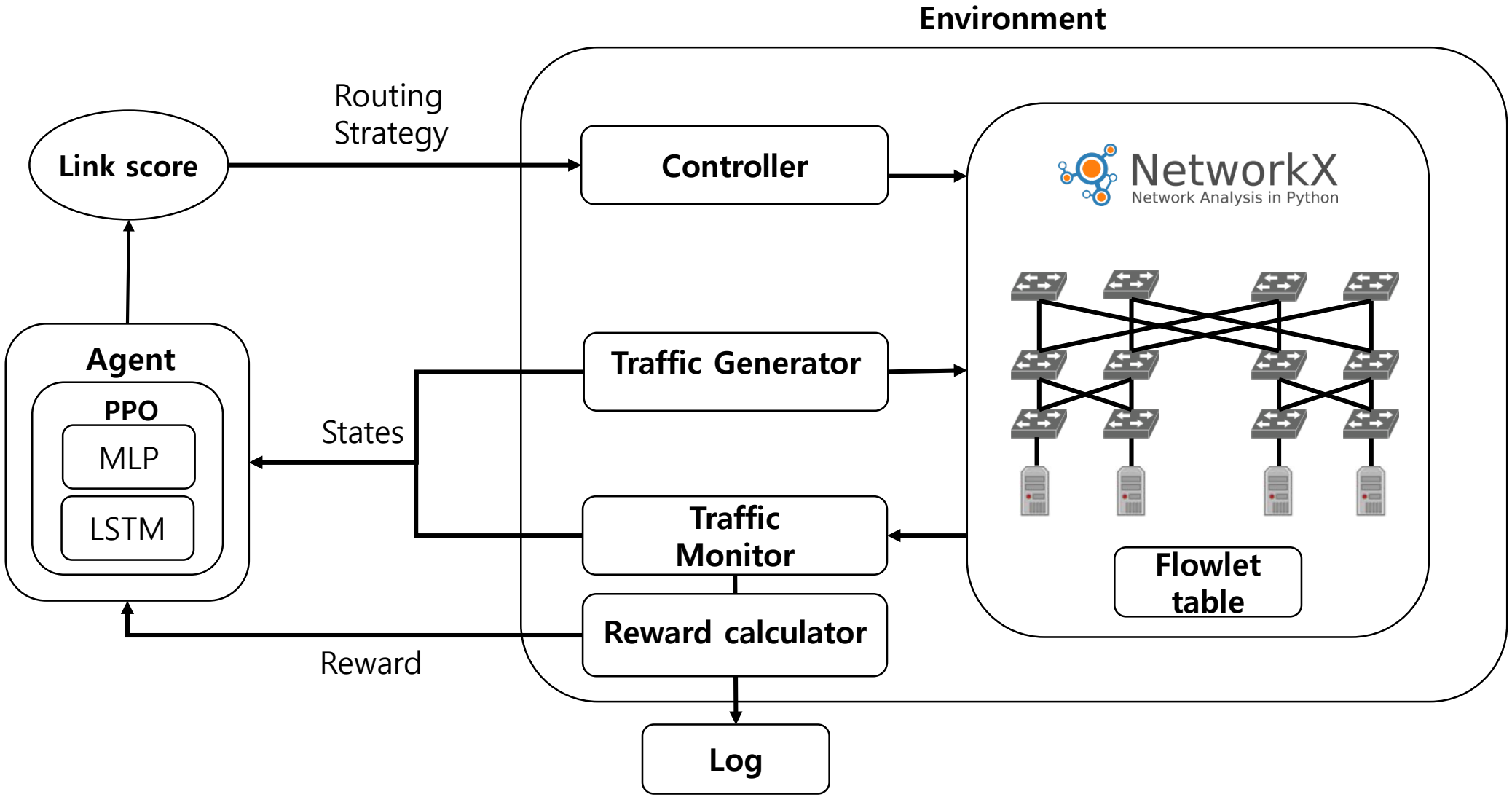
❖ Proximal Policy Optimization algorithm (PPO)

- Actor-Critic Model
  - Actor model : Calculate action
  - Critic model : Evaluate action
  - Use neural network model as a function
    - Multi-layer perceptron (MLP) : 2 layers with 32 node
    - Long-short term memory (LSTM) : 128 lstm cell

- Sample efficient
  - Update as big step as possible with current data
  - Clipping : Don't update too much and loose performance

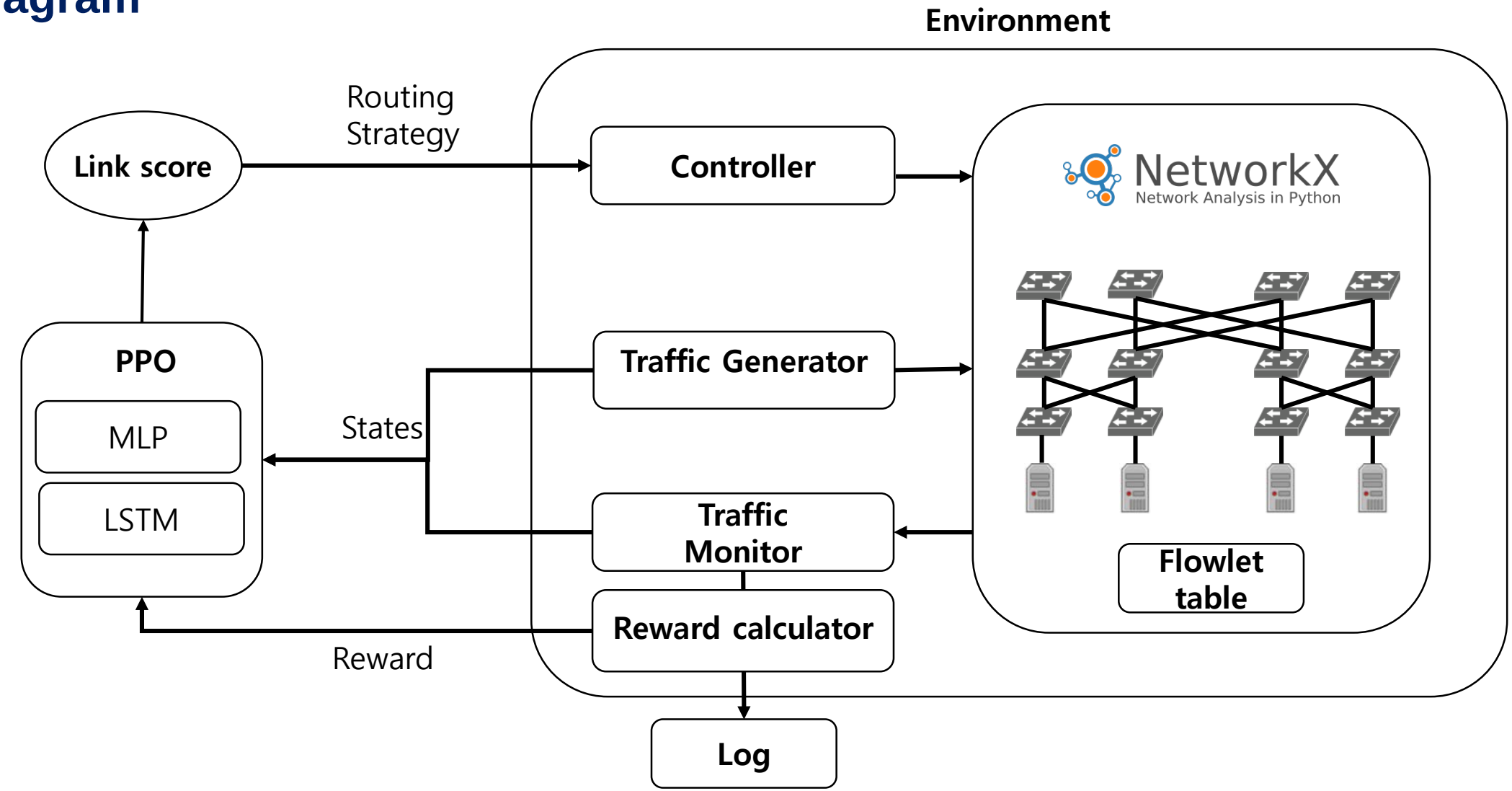


Implementation



Implementation

❖ Diagram



Evaluation

Experimental setup

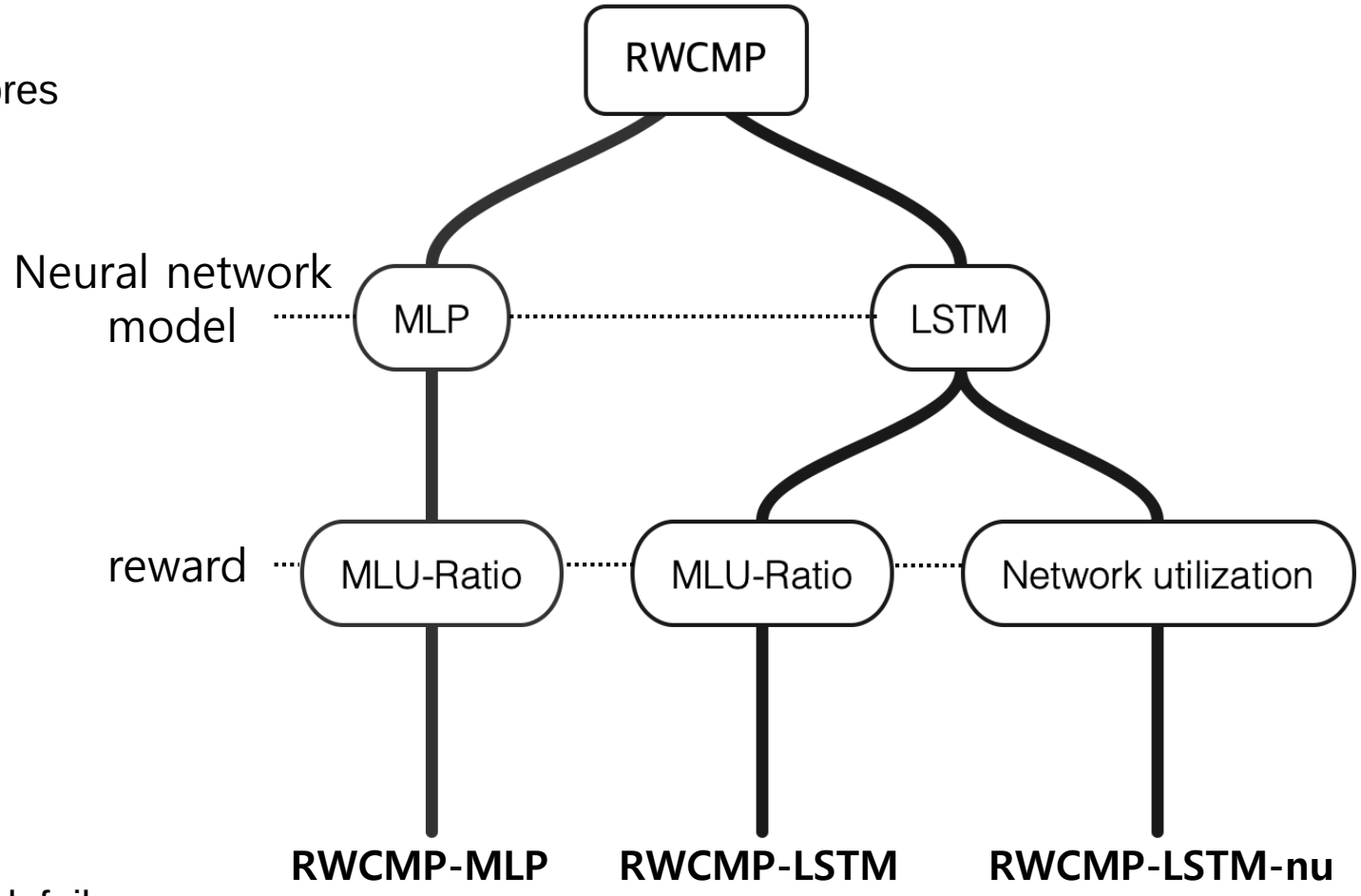
❖ Experiment Environment

- Hardware
  - Intel Xeon Silver 4215 CPU 2.50GHz, 8 cores
  - 64GB RAM
  - RTX 5000 GPU

❖ Testbed setups

- Training setup
  - # Iter : 100K
  - # flows in a iteration : 1000
  - Communication delay : 10ms

- Scenarios
  - Scenario # 1 : Fattree topology
  - Scenario # 2 : Fattree topology with one link failure

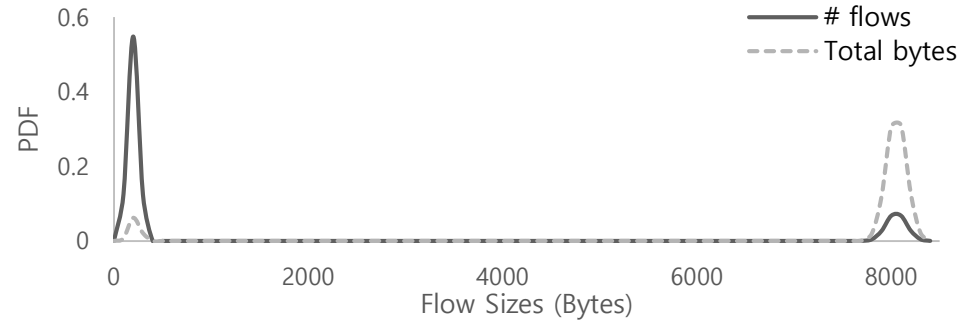


Experimental setup

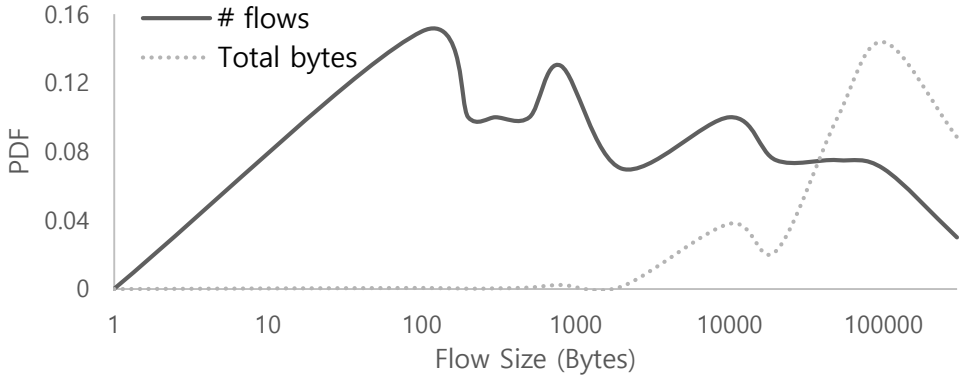
❖ Traffic patterns

- Bimodal
  - Traffic pattern from bimodal distribution
  - Represents small flow and elephant flows both
- DCTCP pattern
  - Traffic pattern from web services
- Facebook pattern
  - Traffic pattern from Facebook

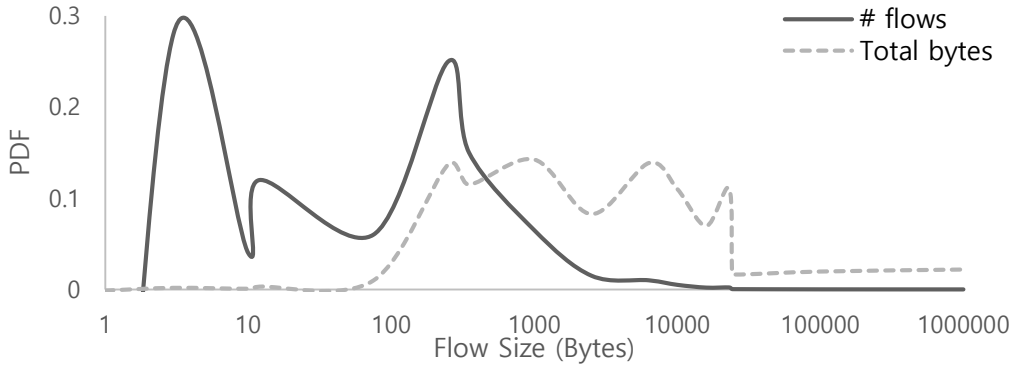
Bimodal



DCTCP

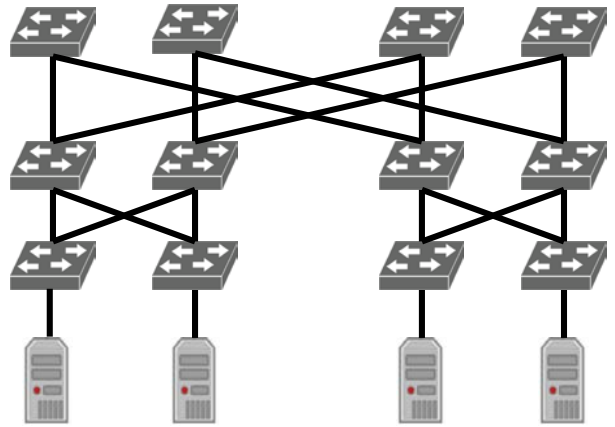


Facebook

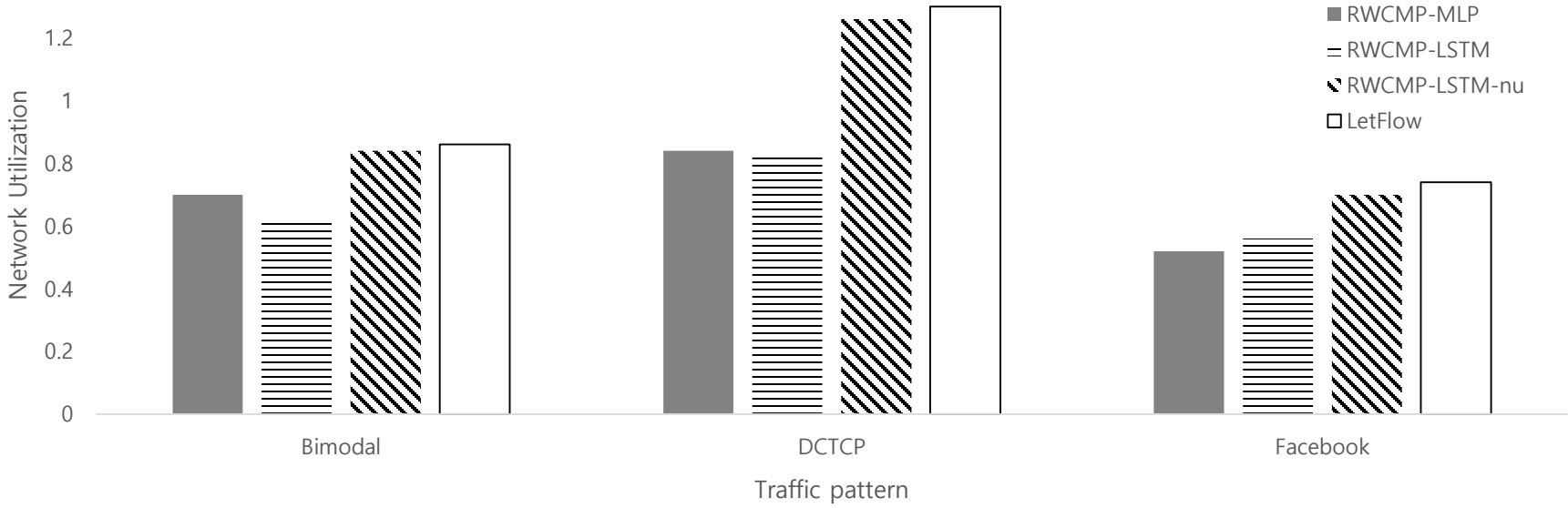


Experiments

❖ Fattree topology

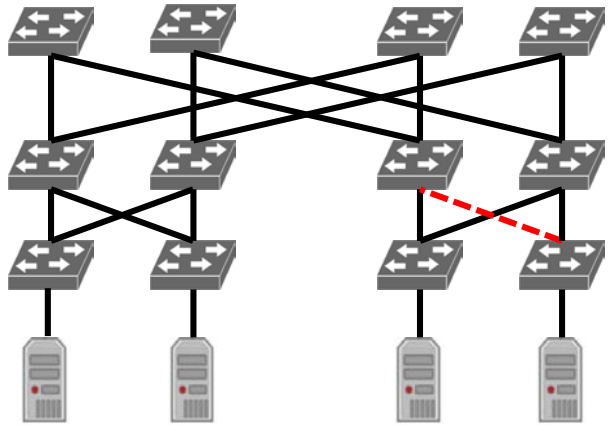


- Generate 1000 flows on on fattree topology
- Measure the network utilization
- Iterate 10 times and compute average network utilization
- Results
  - RWCMP algorithm shows 3~33% lower network utilization than traditional load balancing algorithm
  - Using MLU-ratio reward shows 26~30% lower network utilization than using network utilization reward
  - RWCMP-lstm shows the lowest network utilization (average 0.66 MLU)

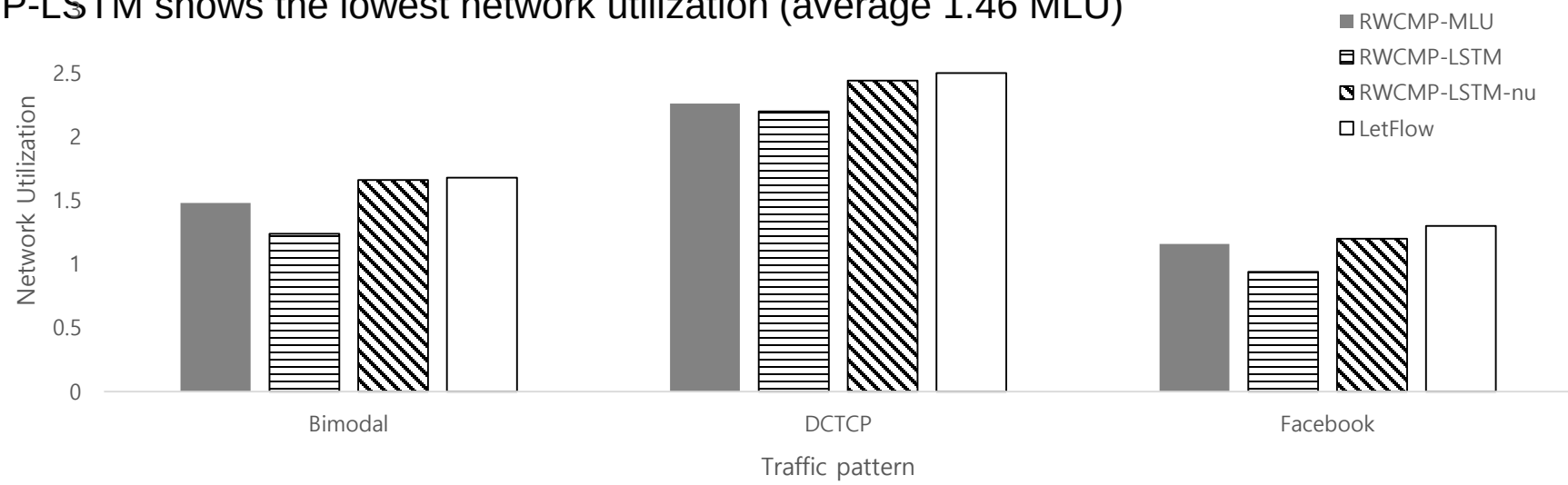


Experiments

❖ Link failure



- Generate 1000 flows on a fattree with one link failure
- Measure the network utilization
- Iterate 10 times and compute average network utilization
- Results
  - RWCMP algorithm shows 4~20% lower network utilization than traditional load balancing algorithm
  - Using MLU-ratio reward shows 8%~18% lower network utilization than using network utilization reward
  - RWCMP-LSTM shows the lowest network utilization (average 1.46 MLU)



Discussion

❖ Results

- RWCMP shows lower mean average network utilization than the traditional load balancing algorithm on both symmetry and asymmetry topology
- Using MLU-Ratio reward shows better performance than using network utilization ratio
- Using LSTM model shows better performance than MLP model

❖ Weakness

- Simulated with python simulator
- Asymmetry : Evaluate performance after link failure occurs

Conclusion

Conclusion

❖ Summary

- Current load balancing algorithm uses snapshot of network status to route current traffic optimal
- We propose a RWCMP load balancing algorithm to learn traffic pattern to route optimal
- Experimental results
  - RWCMP with LSTM model and MLU-Ratio reward yields better performance than others
  - RWCMP works well in asymmetry topology

❖ Future work

- Design a robust algorithm to link failure during routing with graph neural network (GNN)
- Validate the proposed method with longer communication delay with controller
- Validate the proposed method with other metrics such as throughput or latency
- Extends testbed to real network

감사합니다