Optimizing Network Digital-Twin Controllers for Internet-of-Things Instrumented Smart Environments

Chih-Chun Wu (chihchunwu0517@gmail.com)

Advisor: Cheng-Hsin Hsu

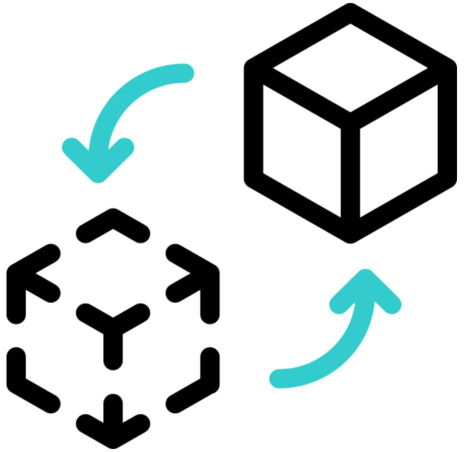
Networking and Multimedia Systems Lab, CS, National Tsing Hua University

NMSL@NTHU

Networking and Multimedia Systems Lab

Outline

- Introduction
- Goal and Challenges
- Related Work
- System Overview
- Problems and Solutions
- Implementations
- Evaluations
- Conclusion & Future Work

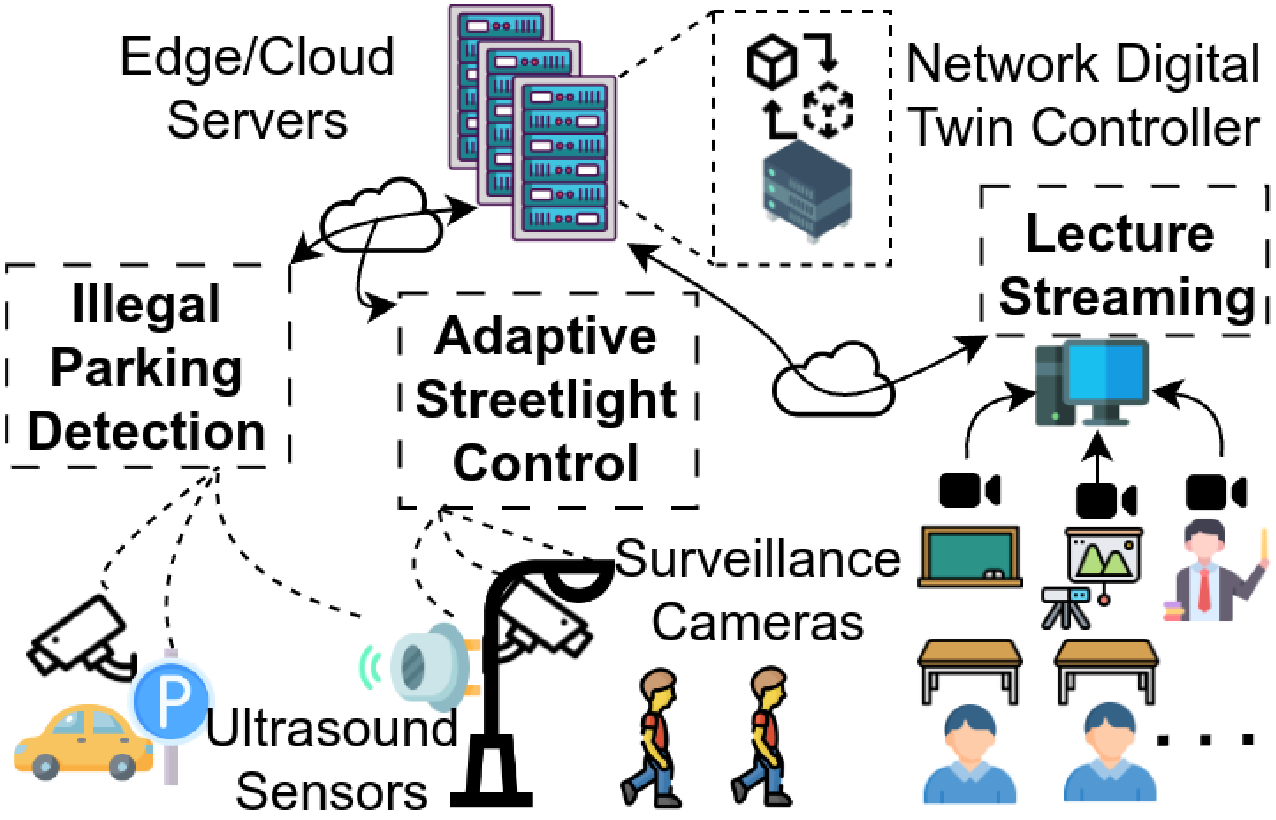


Applications in Smart Environments: Motivations

- Guaranteeing the QoS requirements of each application is no easy task as a large number of physical devices

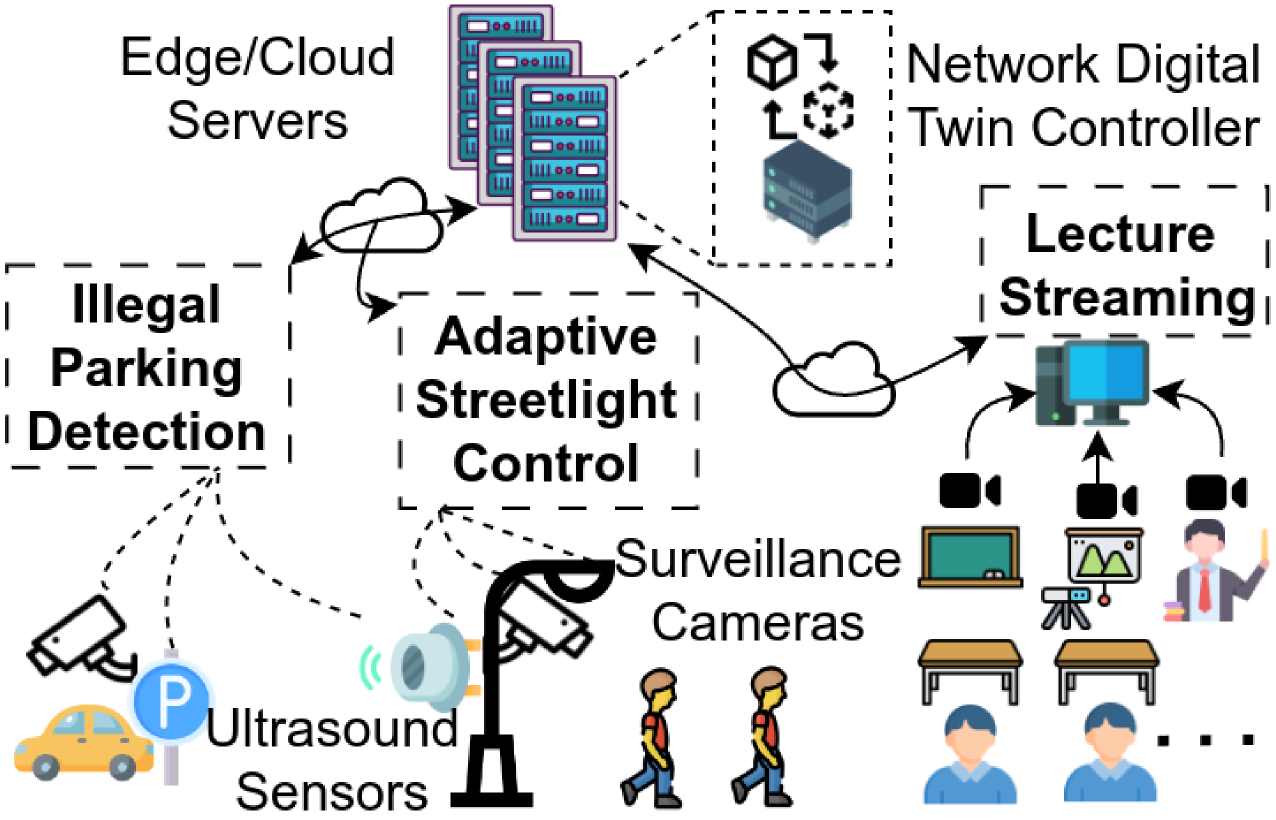
- Administrators face challenges in determining how to upgrade smart environments when the QoS requirements cannot be met

- Example: Manual upgrades for lecture streaming in a classroom network



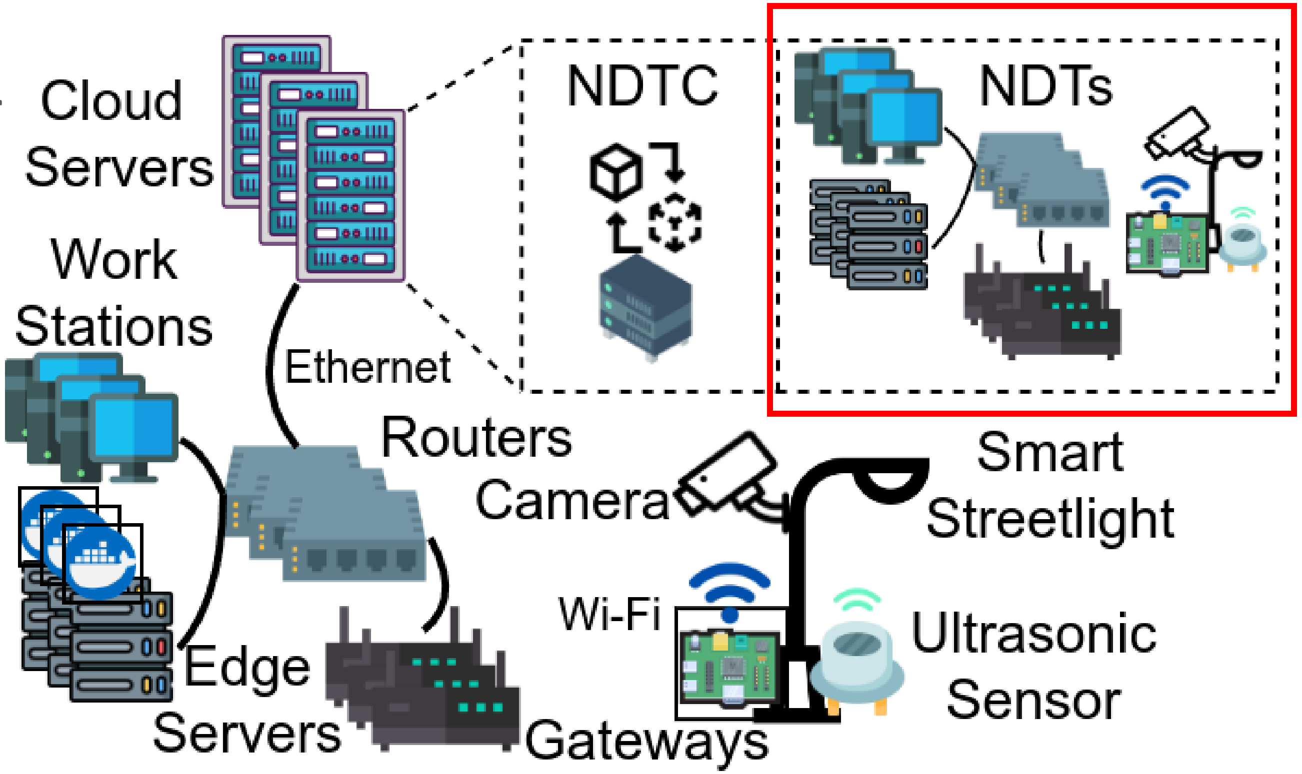
Previous Solution

- Traditional methods rely on trial and error for maintaining and upgrading QoS
  - Error-prone
  - Time-consuming
- High deployment and operational costs due to overlapping monitoring devices
- Need a cost-effective method



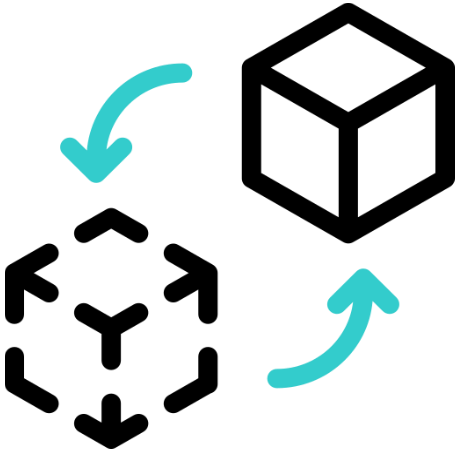
Solution Approach: Utilize Network Digital Twins (NDTs)

- A Digital Twin (**DT**) is a digital replica of a physical entity, referred to as the Physical Twin (**PT**), capturing its state and behavior
- **NDT**: Extends DTs to networked devices and links, creating virtual instances of the network infrastructures
- Provides dynamic, real-time representation of network states, which can be shared by multiple applications
  - Reduce costs
  - Support What-if analysis



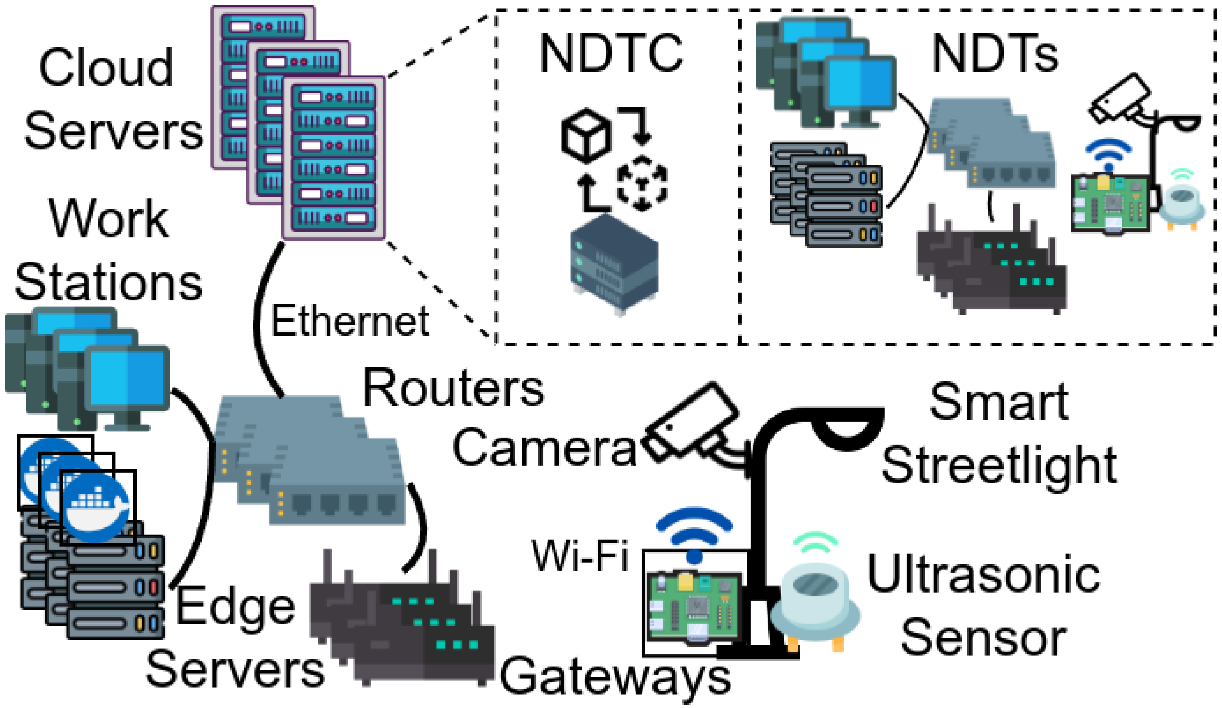
Outline

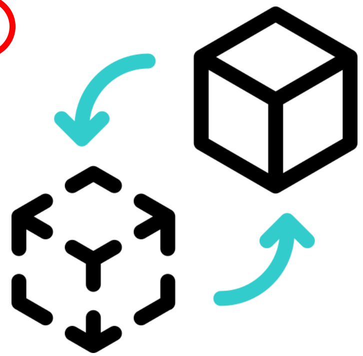
- Introduction
- Goal and Challenges
- Related Work
- System Overview
- Problems and Solutions
- Implementations
- Evaluations
- Conclusion & Future Work



Propose Network Digital Twin Controller (NDTC)

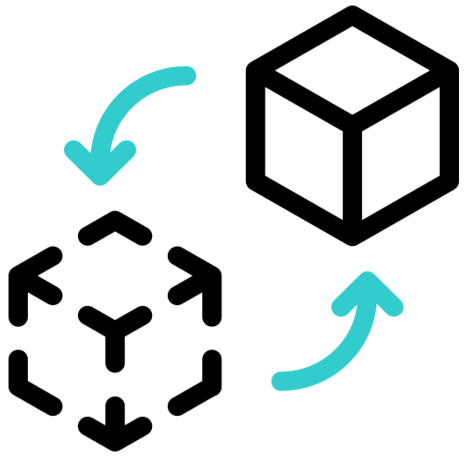
1. Create DT instances as digital replications
2. Synchronize states between DTs and PTs
3. Provide what-if analysis for different settings and configurations

1


2


3


Key Challenges of the NDT Framework: State Synchronization



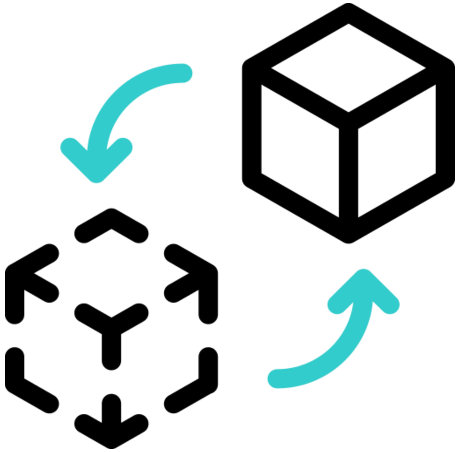
- DTs of workstations and IoT devices in a smart environment must reflect real-time changes in PTs' states
- If update frequency is too low or data granularity is too coarse, DTs may show outdated configurations and sensor readings, affecting management decisions
- Conversely, excessively high frequency or too fine data granularity can cause excessive network traffic and latency
- Update frequency, data granularity, and bandwidth budgets need to be considered !

Key Challenges of the NDT Framework: What-if Analysis

- For lecture streaming, a high accuracy simulator may be needed to predict network bandwidth and ensure smooth streaming. However, this simulator consumes significant computational resources
- For adaptive streetlight control, a simpler queuing theory might suffice, as it requires less accuracy and fewer resources
- QoS prediction accuracy and resource budgets need to be considered !

Key Challenges of the NDT Framework: Trade-off

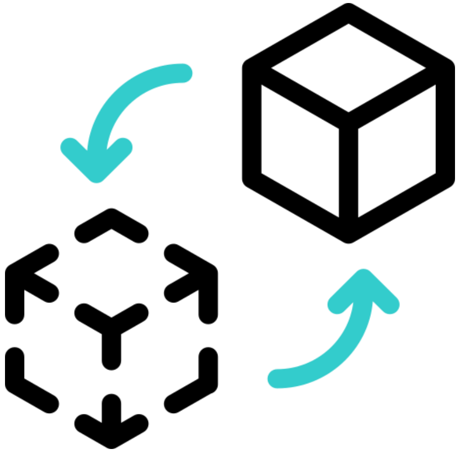
- State synchronization problem:
  - Adjust update frequency, data granularity for state synchronization based on real-time network conditions
  - Objective: **Minimize overall state deviations between DTs and PTs within given bandwidth budgets**



- What-if analysis problem:
  - Choose the what-if analyzer with best QoS prediction accuracy for each query from administrator
  - Objective: **Minimize overall prediction error of queries within given computing resource budgets**

Outline

- Introduction
- Goal and Challenges
- Related Work
- System Overview
- Problems and Solutions
- Implementations
- Evaluations
- Conclusion & Future Work



Network Digital Twin Controller

- Instead of developing NDTCs from scratch, researchers [1,2,3] have extended SDN controllers for NDTs:
  - Improve NDTs' query efficiency [2]
  - Provide a unified interface to create DTs of heterogeneous devices [3]

- No consideration of optimization NDTCs:
  - State synchronization: update frequency and data granularity
  - What-if analysis: QoS prediction accuracy

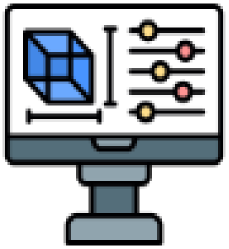
[1] Polverini, Marco, et al. "Digital twin manager: A novel framework to handle conflicting network applications." 2022 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN). IEEE, 2022.
[2] Raj, Deepu Raj Ramachandran, et al. "Building a Digital Twin Network of SDN Using Knowledge Graphs." IEEE Access 11 (2023): 63092-63106.
[3] Hong, Hanshu, et al. "Netgraph: An intelligent operated digital twin platform for data center networks." Proceedings of the ACM SIGCOMM 2021 workshop on network-application integration. 2021.

State Synchronization

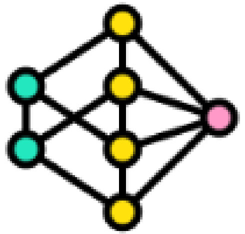
- Adaptive SDN state updates: consider update frequency and bandwidth budget [1,2]
- NDT prioritized synchronization mechanisms [3]
- No consideration of both update frequency and data granularities

[1] Tangari, Gioacchino, et al. "Self-adaptive decentralized monitoring in software-defined networks." IEEE Transactions on Network and Service Management 15.4 (2018): 1277-1291.
[2] Poularakis, Konstantinos, et al. "Learning the optimal synchronization rates in distributed SDN control architectures." IEEE INFOCOM 2019-IEEE Conference on Computer Communications. IEEE, 2019.
[3] Mishra, Shivakant, and Khaled Alanezi. "Towards a scalable architecture for building digital twins at the edge." Proceedings of the Eighth ACM/IEEE Symposium on Edge Computing. 2023.

What-If Analysis

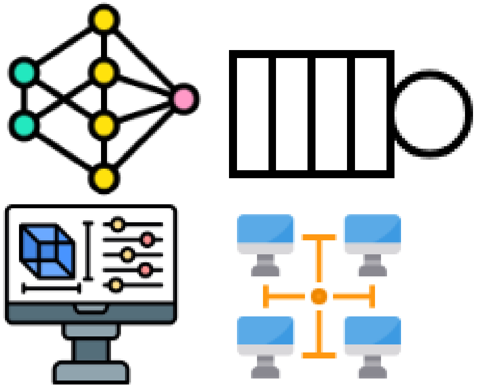


- Simulator-based what-if analyzer [1,2]: simulate network traffic under different configurations



- ML-based what-if analyzer [3,4]: use Graph Neural Network (GNN) to predict QoS metrics

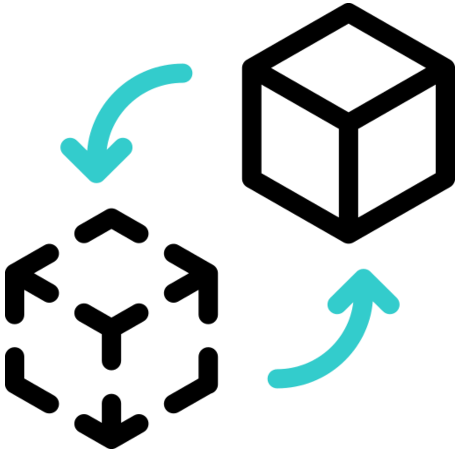
- Only provide a homogeneous what-if analyzer
  - Our NDTC supports multiple what-if analyzers with varying accuracy and resource demands
  - The selections of what-if analyzers are made by our what-if analysis algorithm



[1] Polverini, Marco, et al. "A Digital Twin based Framework to Enable "What-If" Analysis in BGP Optimization." NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium. IEEE, 2023.
[2] Wieme, Jorg, Mathias Baert, and Jeroen Hoebeke. "Managing a QoS-enabled Bluetooth Mesh network using a Digital Twin Network: An experimental evaluation." Internet of Things 25 (2024): 101023.
[3] Rusek, Krzysztof, et al. "Unveiling the potential of graph neural networks for network modeling and optimization in SDN." Proceedings of the 2019 ACM Symposium on SDN Research. 2019.
[4] Yu, Peng, et al. "Digital twin driven service self-healing with graph neural networks in 6g edge networks." IEEE Journal on Selected Areas in Communications (2023).

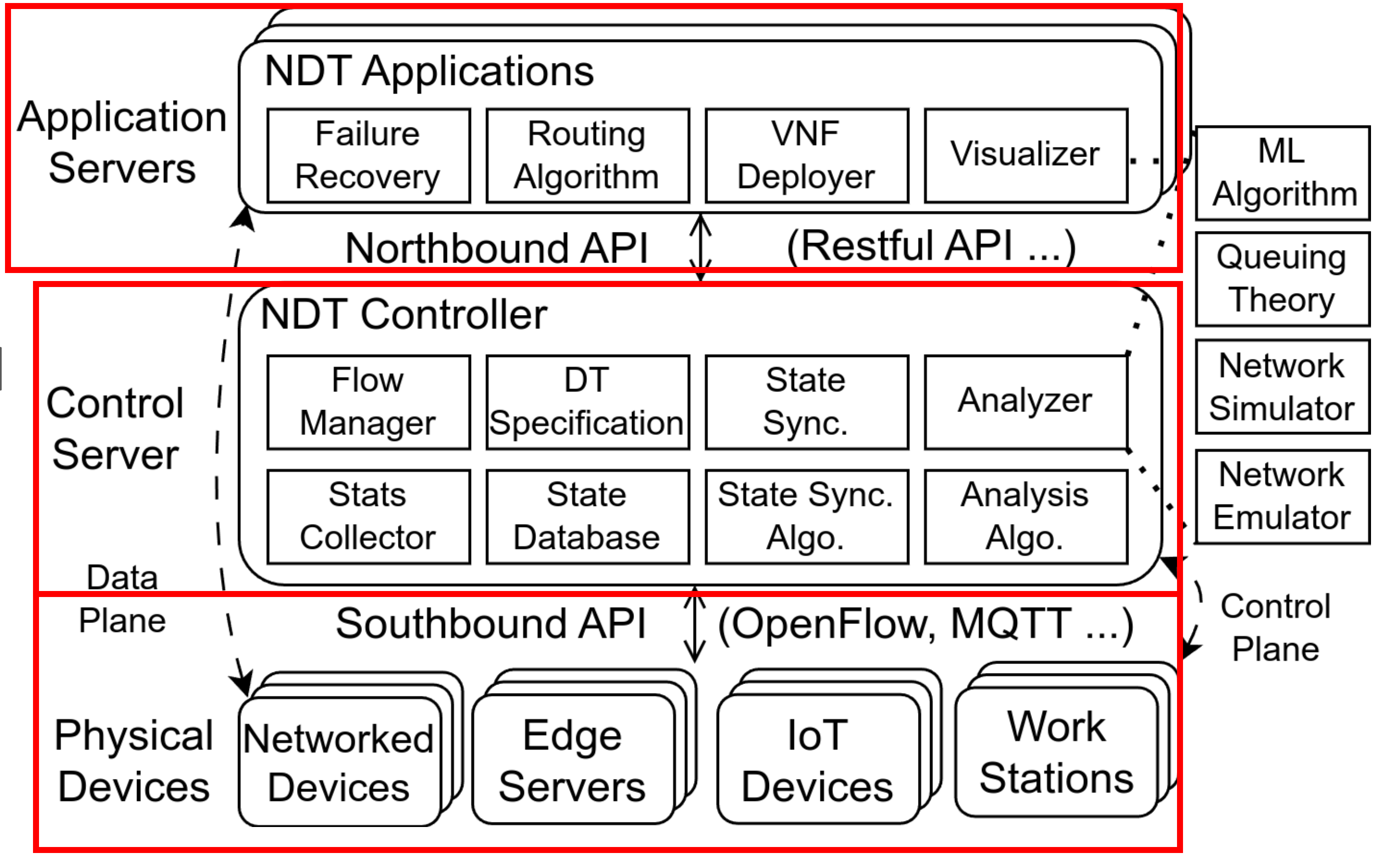
Outline

- Introduction
- Goal and Challenges
- Related Work
- System Overview
- Problems and Solutions
- Implementations
- Evaluations
- Conclusion & Future Work



Network Digital Twin Controller

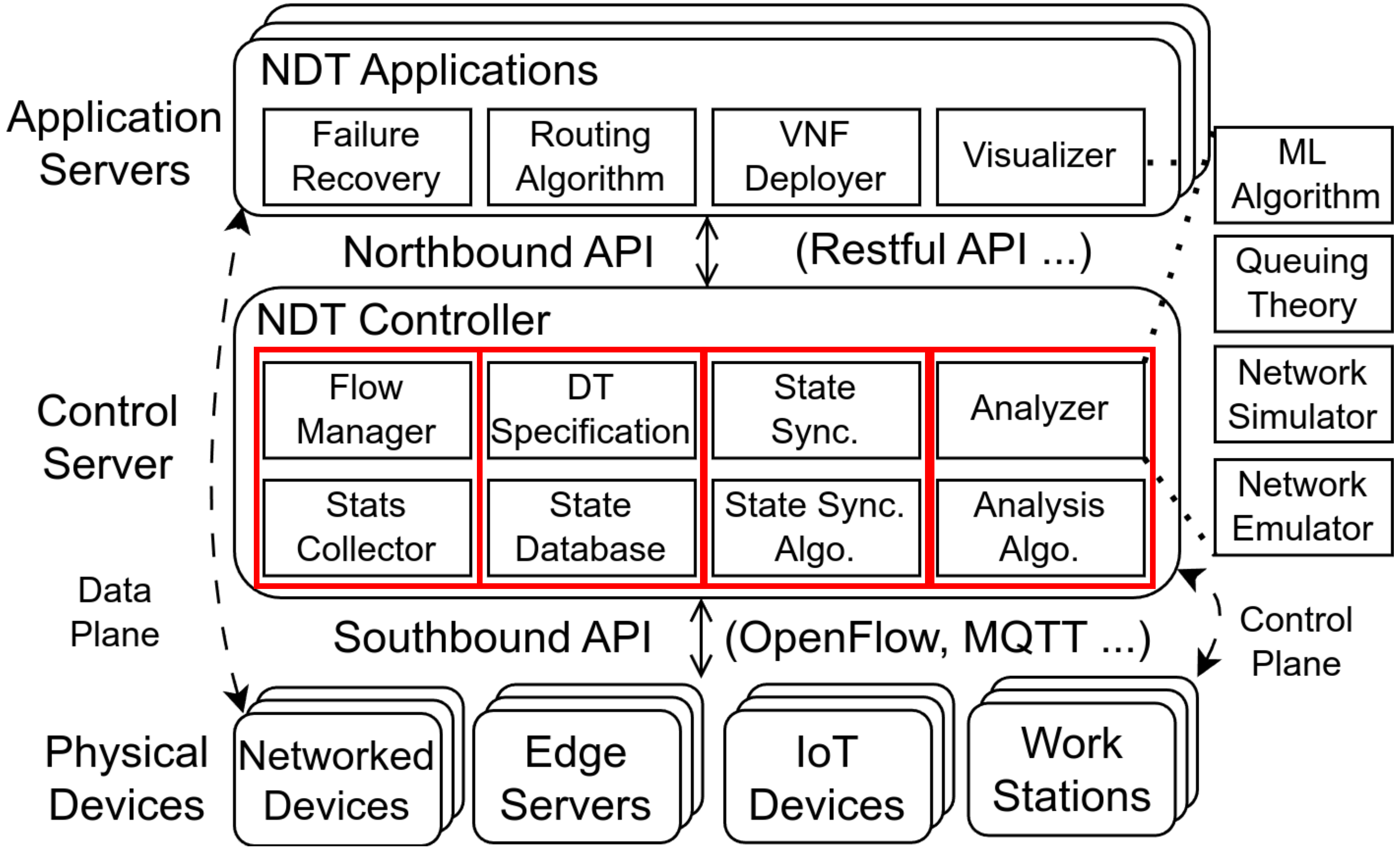
- Built on an open-source SDN controller
- Provides
  - Southbound API: for physical devices to update their states and receive dictated actions
  - Northbound API: exchange data with NDT applications



System Overview

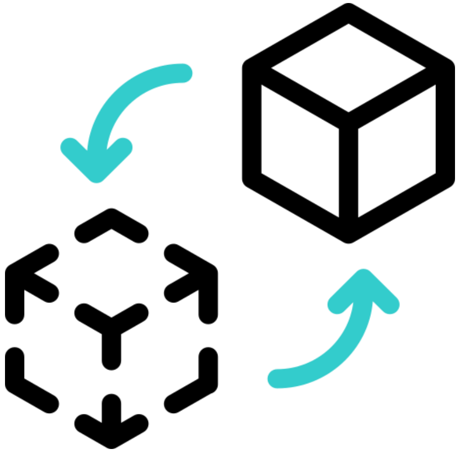
Network Digital Twin Controller: SDN & NDT Modules

- Flow Manager
- Stats Collector
- **DT Specification**
- **State Database**
- **State Synchronizer**
- **State Synchronization Algorithm**
- **What-if Analyzer**
- **What-if Analysis Algorithm**



Outline

- Introduction
- Goal and Challenges
- Related Work
- System Overview
- Problems and Solutions
- Implementations
- Evaluations
- Conclusion & Future Work



State Synchronization Problem

- n : n-th PT/DT
- t : time slot
- k : k-th state
- s_d : state of DT
- s_p : state of PT

- α : weight of states
- β : weight of EWMA

- **State deviation :**

① The state difference between DT and PT

② ③

$$\theta(n,t) = \beta \sum_{k=1}^{K_n} \alpha_{n,k} \frac{|s_d(n,t,k) - s_p(n,t,k)|}{s_p(n,t,k)} + (1-\beta)\theta(n,t-1).$$

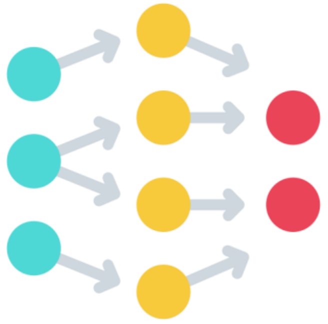
Calculate the overall state deviation using different weights

Employ Exponentially Weighted Moving Average (EWMA) [1] to filter out high-frequency noise

- The expected state deviation under any update frequency and data granularity needs to be predicted

[1] Edalat, Yalda, Jong-Suk Ahn, and Katia Obraczka. "Smart experts for network state estimation." IEEE Transactions on Network and Service Management 13.3 (2016): 622-635.

Employ Online Machine Learning (OML) Algorithms



- Build a prediction model $\tilde{\theta}(f_n, z_n)$

$$\theta(n,t) \simeq \tilde{\theta}(f_n, z_n)$$

- Upon receiving an update message, the computed $\theta(n,t)$ is sent into an OML algorithm to refine $\tilde{\theta}(f_n, z_n)$

- Several OML algorithms have been proposed [2], and we empirically compare the representative ones in the evaluation chapter

[1] Ridwan, Mohammad Azmi, et al. "Applications of machine learning in networking: a survey of current issues and future challenges." IEEE access 9 (2021): 52523-52556.
[2] Benczúr, András A., Levente Kocsis, and Róbert Pálovics. "Online machine learning in big data streams." arXiv preprint arXiv:1802.05872 (2018).

Minimize State Deviation under Bandwidth Budgets

② Predicted state deviation

B	: bandwidth of NDTC
B_n	: bandwidth of PT n
$m_{n,k}$	: data size of PT n's state k

① Choose **update frequency** and **data granularity** for all PTs

$$\underset{\{f_n, z_n \mid \forall n\}}{\text{minimize}} \sum_{n=1}^{N} \tilde{\theta}(f_n, z_n)$$

③ Bandwidth budgets for each PTs

$$\text{s.t.: } f_n \times \sum_{k=1}^{z_n} m_{n,k} \leq B_n, \forall n \in \{1, 2, \ldots, N\};$$

④ Overall Bandwidth budgets for NDTC

$$\sum_{n=1}^{N} f_n \times \sum_{k=1}^{z_n} m_{n,k} \leq B.$$

Our Algorithms for State Synchronization Problem

- **Optimal Update (OU) algorithm:**
  - Use generic solver (CPLEX) to solve the formulation optimally
  - Take too long to complete for larger problems

- **Gradient-Driven Update (GU) algorithm:**
  - Compute gradient of update frequency and data granularity: the state deviation differences per unit bandwidth change

Two Phases of GU Algorithm

- **Per-PT phase**: for all PTs,
  - Calculate the gradient of update frequency and data granularity
  - Adjust update frequency or data granularity based on the calculated gradient
  - Verify the PT's bandwidth budget

Use minimal bandwidth increases to achieve maximal reductions in state deviation

- **Overall phase**: If the total bandwidth exceeds the overall bandwidth budget:
  - Calculate gradient of update frequency and data granularity of all PTs
  - Adjust update frequency or data granularity of one PT based on the calculated gradient
  - Verify the overall bandwidth budget

Use minimal state deviation increases to achieve maximal reductions in bandwidth

What-If Analysis Problem

- **Prediction error from what-if analyzer :** ① The difference between predicted value and ground truth value of QoS metric

$$e(q,w) = \frac{1}{E}\sum_{e=1}^{E}\left\|\frac{p_e - \tilde{p}_e}{p_e}\right\|$$

② Calculate the average prediction error of all QoS metrics

p_e : predicted value

$\tilde{p}_e$: ground truth value

Minimize Predicted Error under Computing Time Budgets

② Overall Prediction Error

① Choose what-if analyzer of each queries

$$\underset{\{x_{q,w} | \forall q,w\}}{\text{minimize}} \sum_{q=1}^{Q} \sum_{w=1}^{W} e(q,w) \times x_{q,w}$$

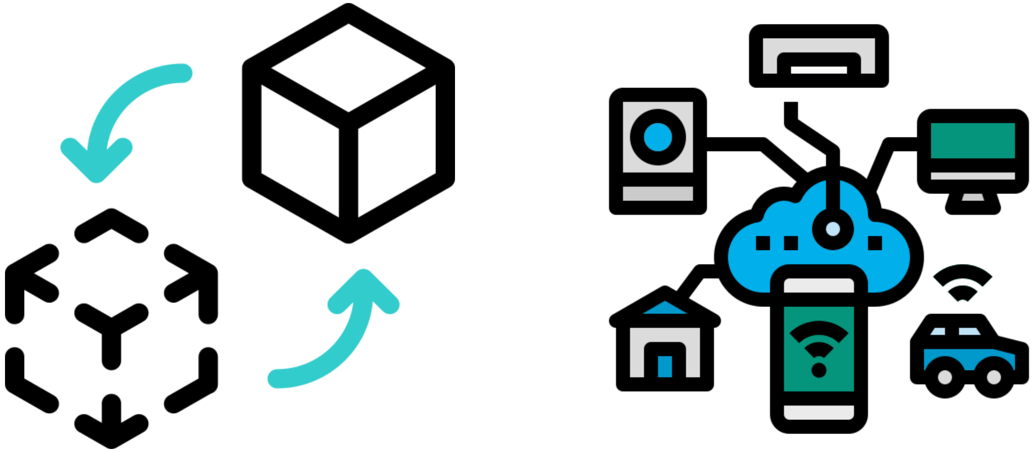
$$\text{s.t.:} \sum_{q=1}^{Q} \sum_{w=1}^{W} c(q,w) \times x_{q,w} \leq C.$$

③ Overall Computing time budgets

- **Optimal Selection (OS) algorithm:** Use generic solver (CPLEX) to solve the formulation optimally

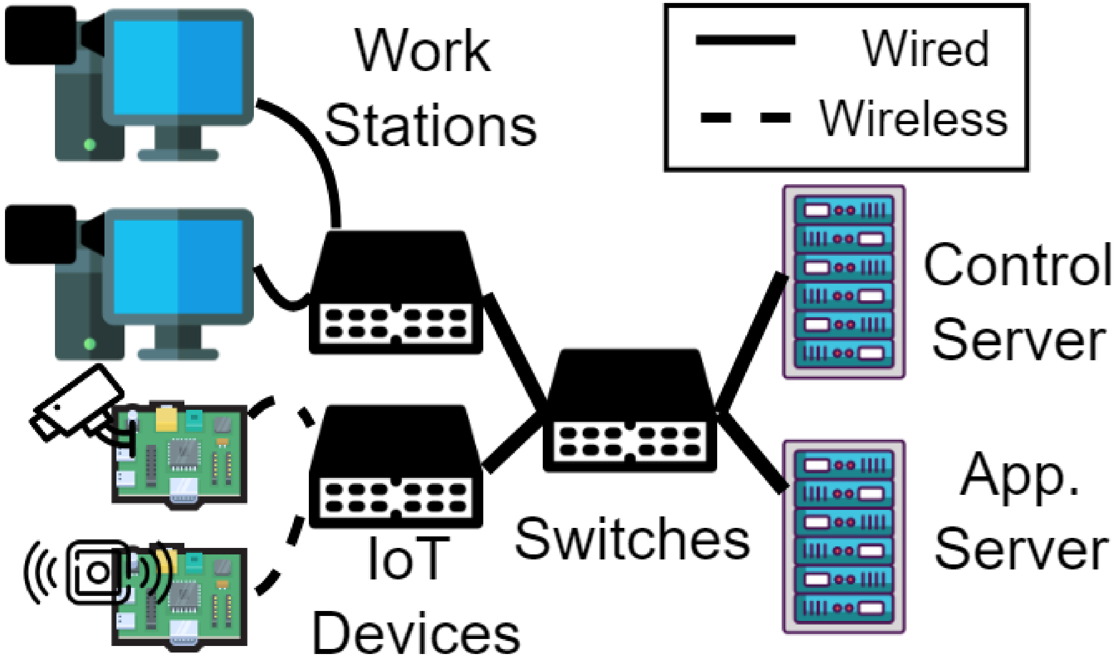
Outline

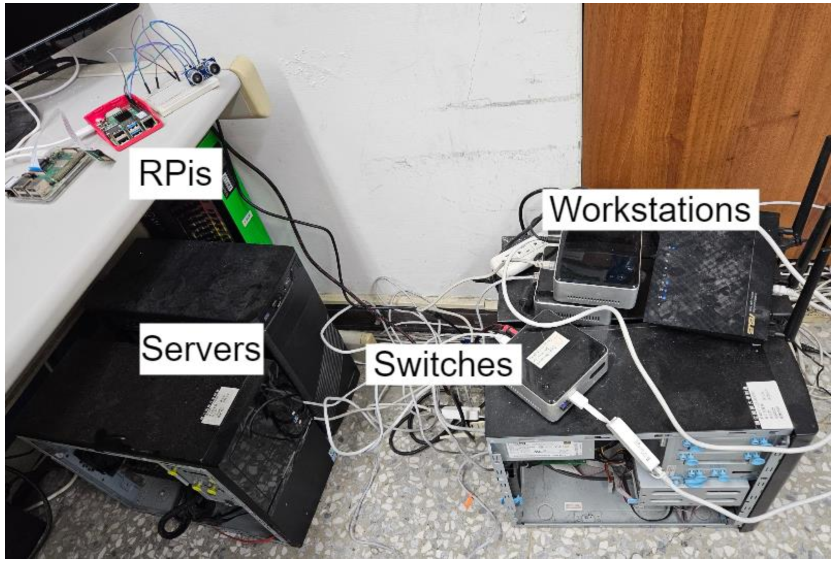
- Introduction
- Goal and Challenges
- Related Work
- System Overview
- Problems and Solutions
- Implementations
- Evaluations
- Conclusion & Future Work



Our Testbed

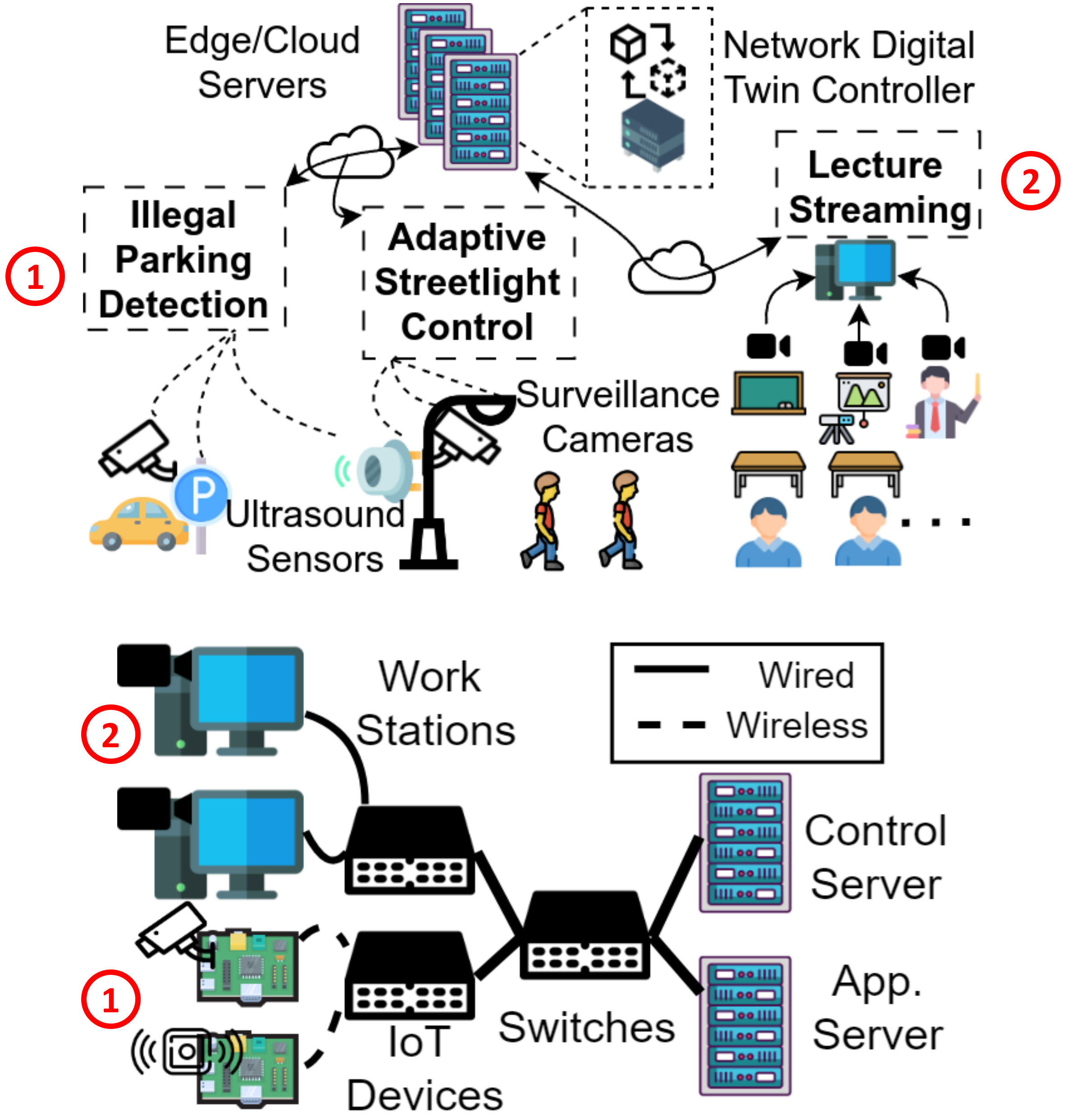
- We implement NDTC based on the open-source **Ryu** SDN controller, running on the control server
- Seven networked devices (work stations, IoT devices, and switches) and two servers are connected by wired and wireless networks
- The network switches are based on the open-source **OpenvSwitch**



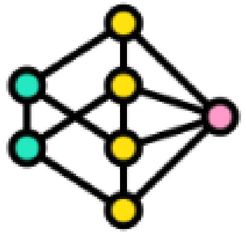


Applications in Testbed

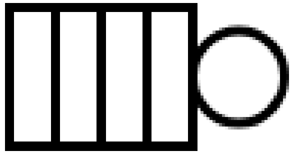
- **Illegal parking detection and adaptive streetlights**: two IoT devices (RPi) stream camera video and ultrasonic sensor distance data
- **Lecture streaming**: two work stations stream two camera feeds of the blackboard and instructor
- All videos are encoded in H.264 at 10 Mbps, and the time-series ultrasound distances occupy 0.5 kbps



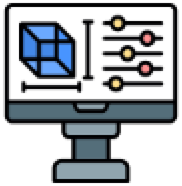
Implementations of What-If Analyzers



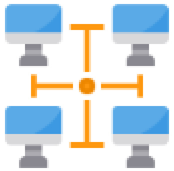
- **ML algorithm**: employ a recent GNN [1]



- **Queuing theory**: model each network link as an M/M/1 queue using the SimPy library



- **Network simulator**: adopt an NS-3 simulator



- **Network emulator**: integrate data flows from our testbed with an NS-3 simulator

[1] Rusek, Krzysztof, et al. "Unveiling the potential of graph neural networks for network modeling and optimization in SDN." *Proceedings of the 2019 ACM Symposium on SDN Research*. 2019

Baseline Algorithms

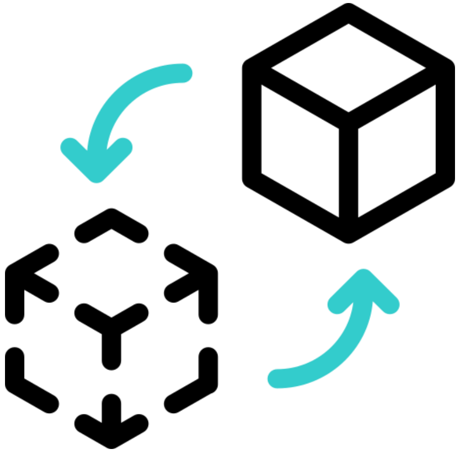
- Two problems were never considered in the literature

- State synchronization problem:
  - Alternative Update (AU) algorithm
  - Random Update (RU) algorithm

- What-if analysis problem:
  - Alternative Selection (AS) algorithm
  - Random Selection (RS) algorithm

Outline

- Introduction
- Goal and Challenges
- Related Work
- System Overview
- Problems and Solutions
- Implementations
- Evaluations
- Conclusion & Future Work



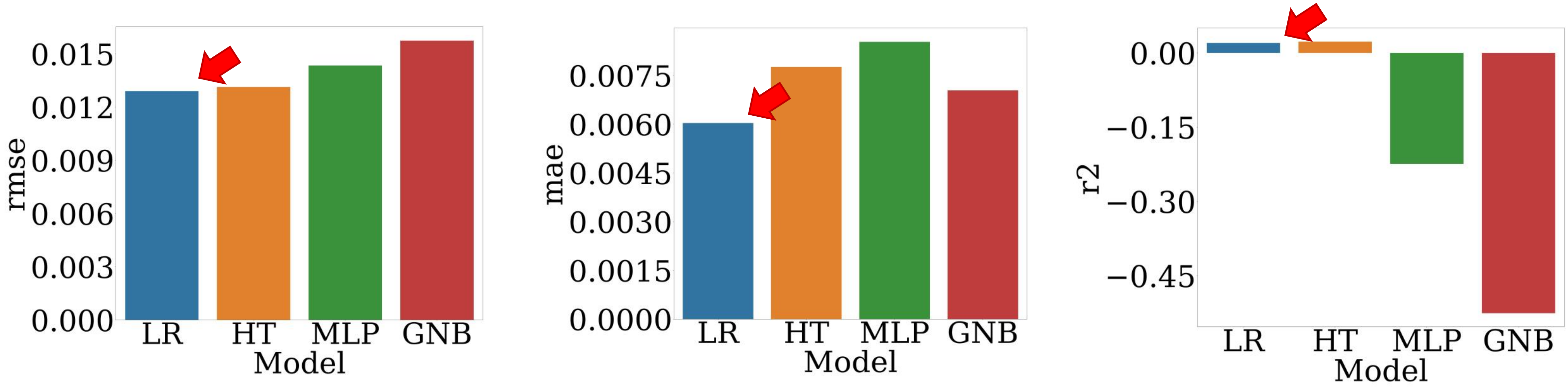
Model Deviation: OML-Based State Deviation Model

- Consider four OML algorithms [1]:
  - Linear Regression (LR)
  - Hoeffding Tree (HT)
  - Multi-layer Perceptron (MLP)
  - Gaussian Naive Bayes (GNB)
- For each device, 1000 state measurements are taken and randomly divided into 80/20 for training/testing
  - Update frequency (f) : {0.1, 0.5, 1, 3, 10} Hz
  - Data granularity (z) : {4, 9, 14}, where the total number of states is 14
  - Bandwidth budget (Bn): {0.5, 1, 2, 4} Mbps

[1] Benczúr, András A., Levente Kocsis, and Róbert Pálovics. "Online machine learning in big data streams." *arXiv preprint arXiv:1802.05872* (2018).

LR Algorithm Constantly Outperforms Other OML Algorithms

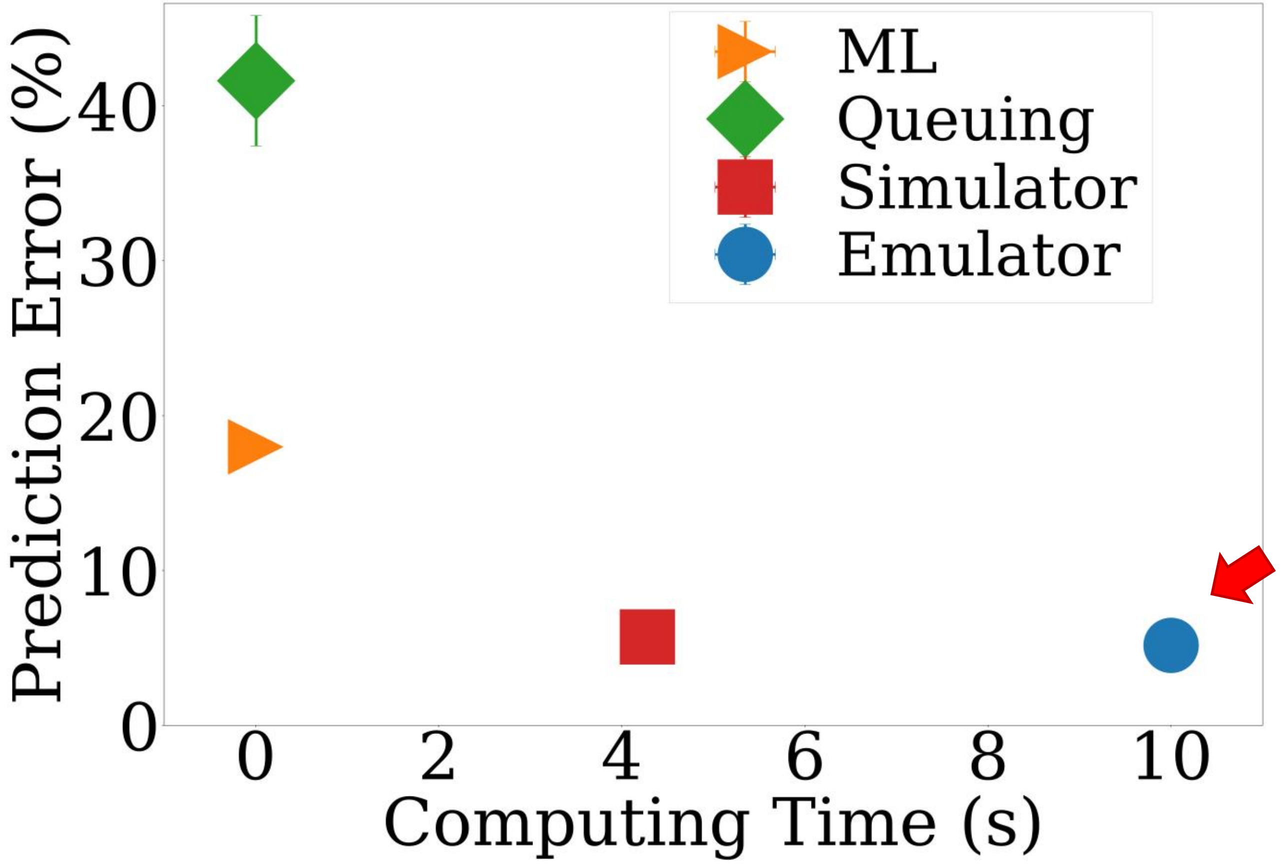
- LR algorithm outperforms at least 4.3% in RMSE, 3.3% in MAE, and 35.5% in R2



- We employ the LR algorithm for the state deviation model

Model Deviation: What-If Analyzers

- Carry out 100 measurements under the workloads generated by the three smart-environment applications
- Trade-off between prediction error and computing time
- The network emulator achieves the smallest prediction error but consumes the most computing time



Experiment Setup

- State synchronization problem:
  - Control server bandwidth budget (B): {5, 10, 15, **30**} Mbps
  - Networked device bandwidth budget (Bn): {0.5, 1, 2, **4**} Mbps
  - Repeat each experiment for 10 runs, where each run lasts for 10 seconds

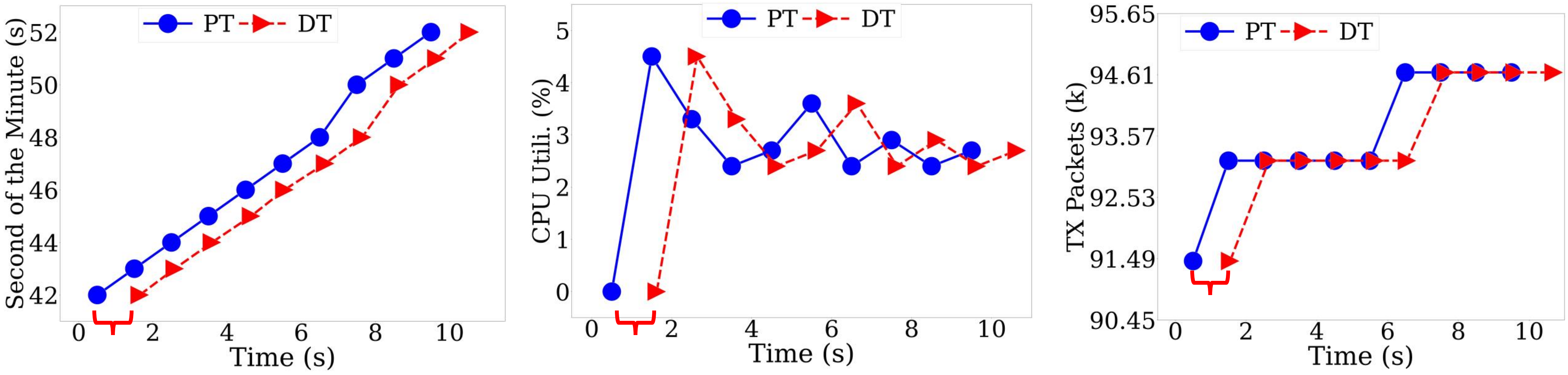
- What-if analysis problem:
  - Number of queries (Q): {15, **20**, 25, 30}
  - Computing time budget (C): {60, **120**, 240, 480} seconds
  - Repeat each experiment for 10 runs

Metrics

- State deviation: θ(·) is computed whenever a state update message is received
- Prediction error: e(·) of the queries from the administration
- Running time of the optimization algorithms
- Memory utilization of the application server
- CPU utilization of the application server
- Control message throughput at the control server

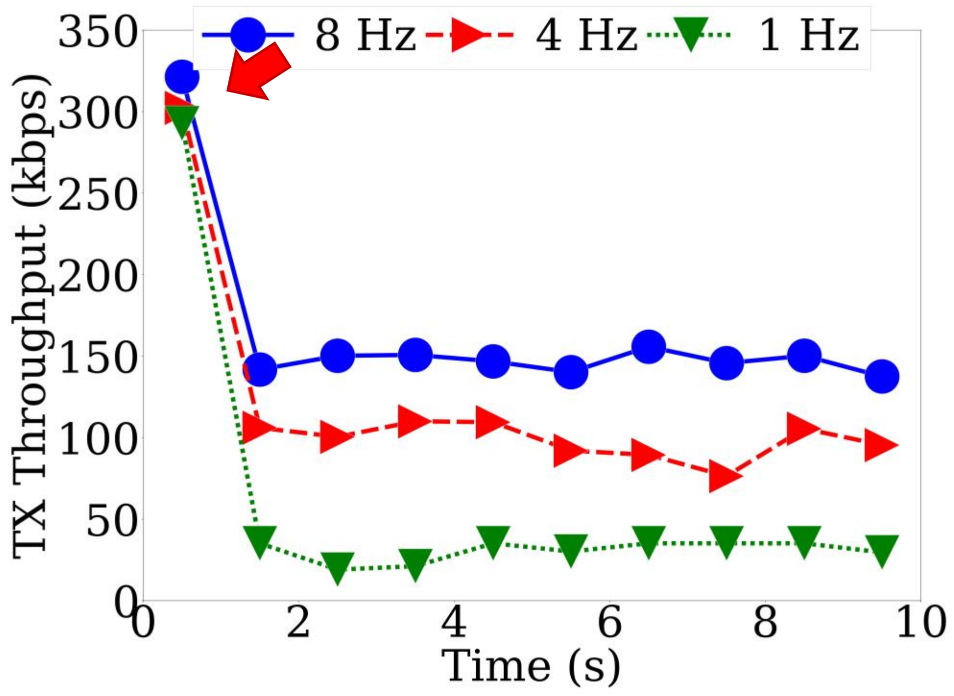
PT/DT States from an IoT Device, a Workstation, and a Network Switch

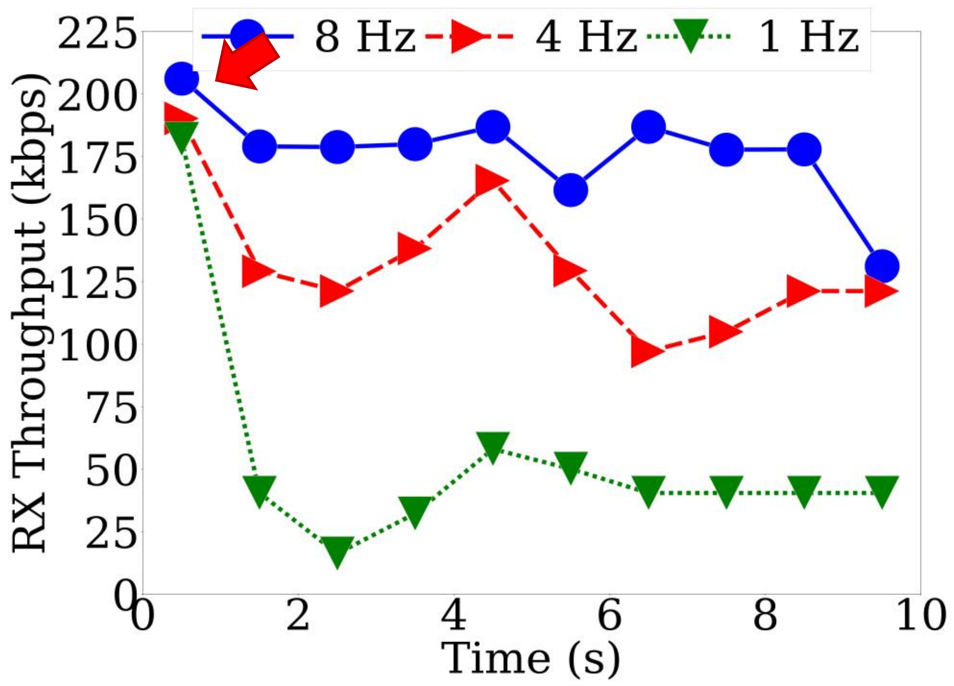
- The DT states are updated every second, which is the chosen update frequency



Without Incurring Excessive Overhead

- The control message throughput at the server is manageable in modern networks





The QoS Prediction and Computing Time of Individual What-if Analyzers

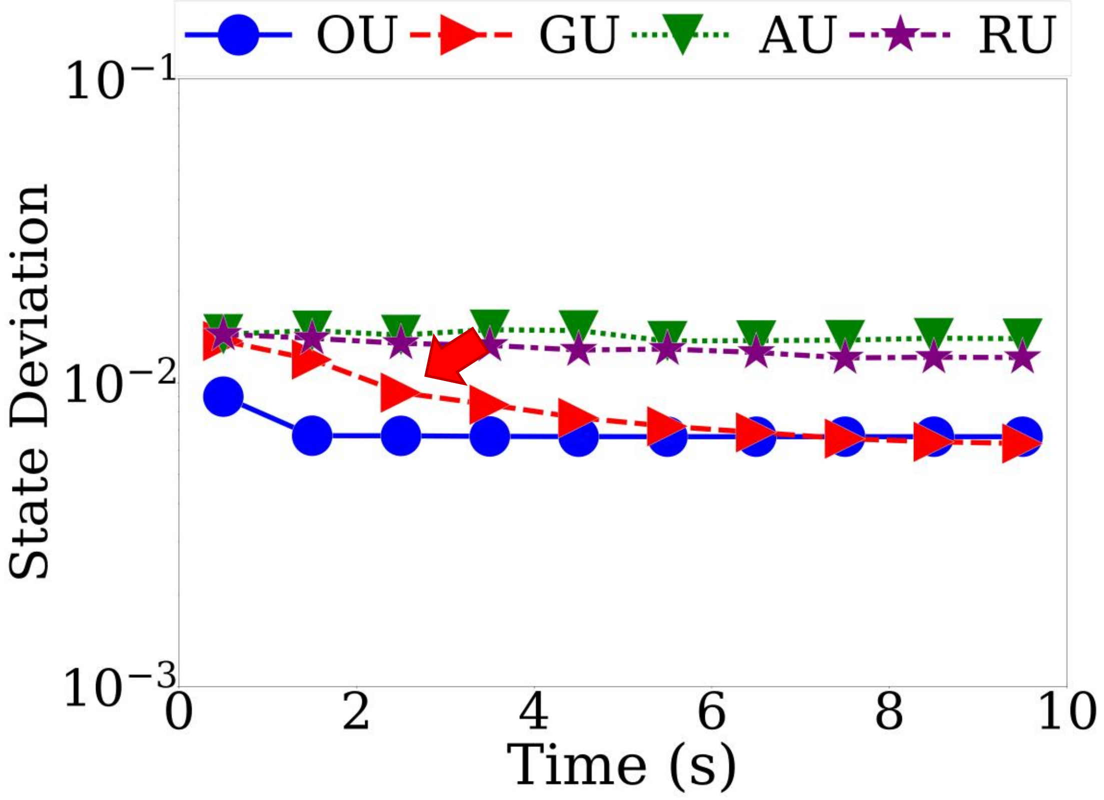
- For a sample what-if question: "What are the expected QoS measurements if we reduce the bitrate of all video streams from 10 to 5 Mbps?"

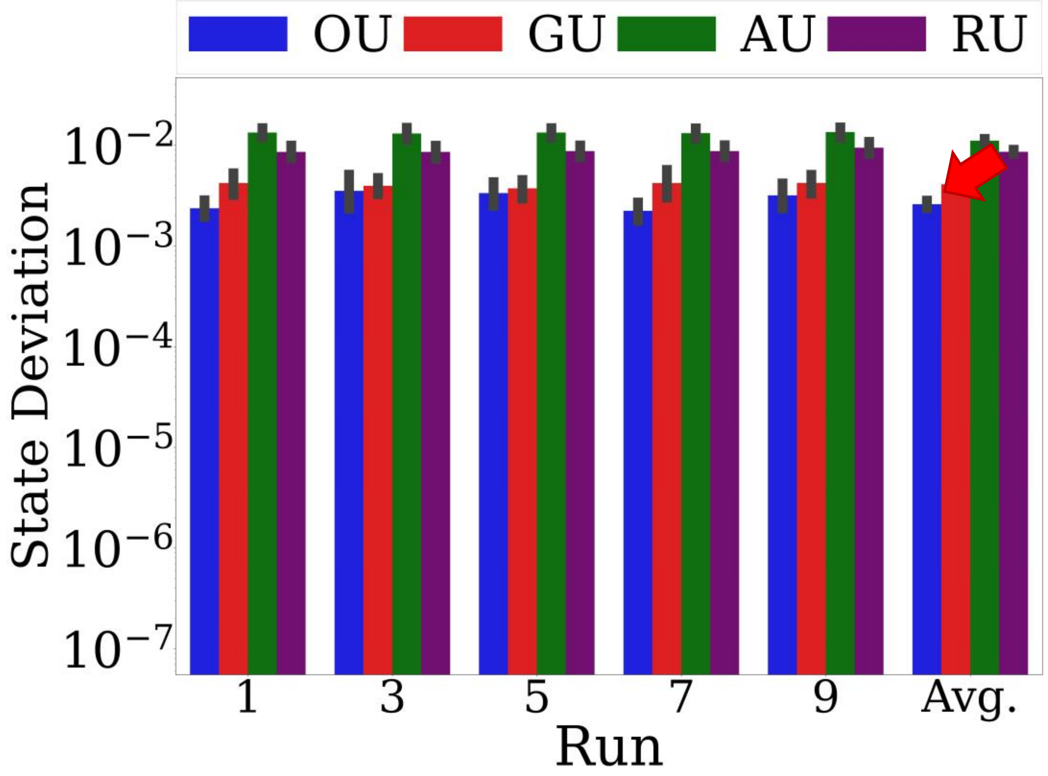
Metrics	Pred. Delay (ms)	Pred. Jitter (ms)	Comp. Time (ms)
ML	19.07	16.88	2.38
Queuing	43.50	40.40	5.16
Simulator	10.32	0.34	4338
Emulator	11.06	0.10	10,000

- These sample results confirm our NDTC works well

OU and GU Reduce the Expected State Deviation

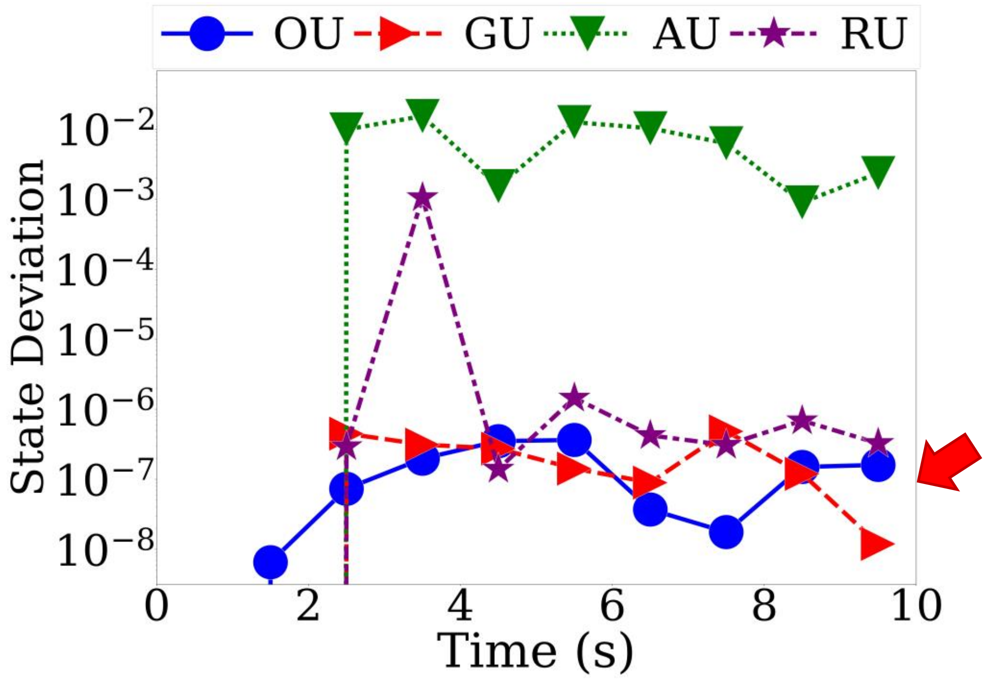
- OU leads to the lowest state deviation, up to 8.39 times smaller than the baseline algorithms
- GU also results in a smaller state deviation than the baseline algorithm

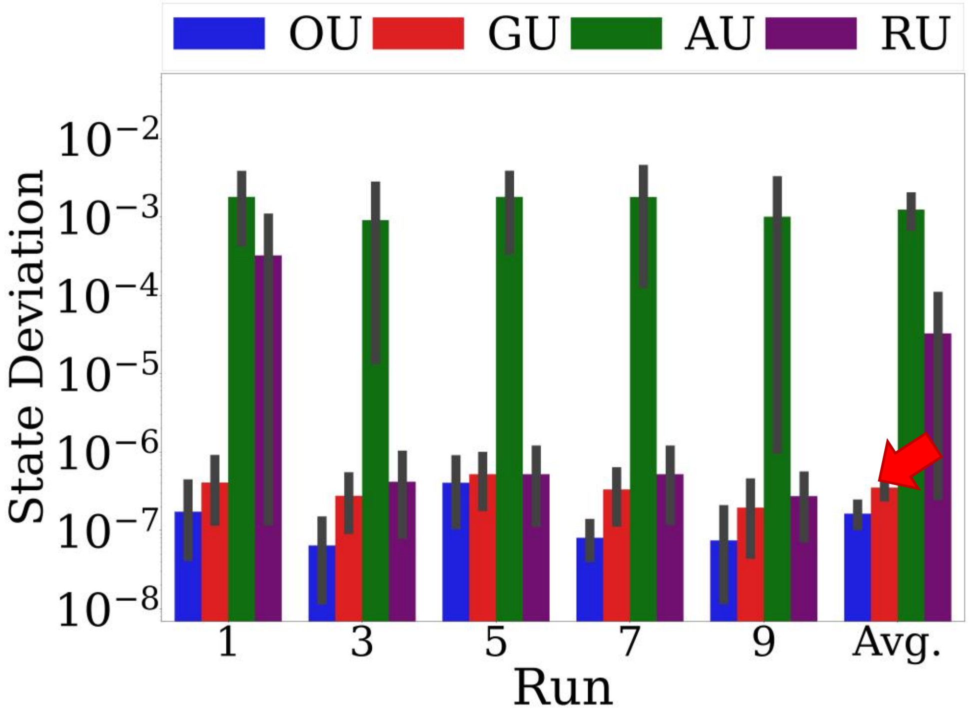




OU and GU Reduce the Actual State Deviation

- OU and GU outperform the baseline algorithms at least 99.49% and 98.92% reductions in the state deviation

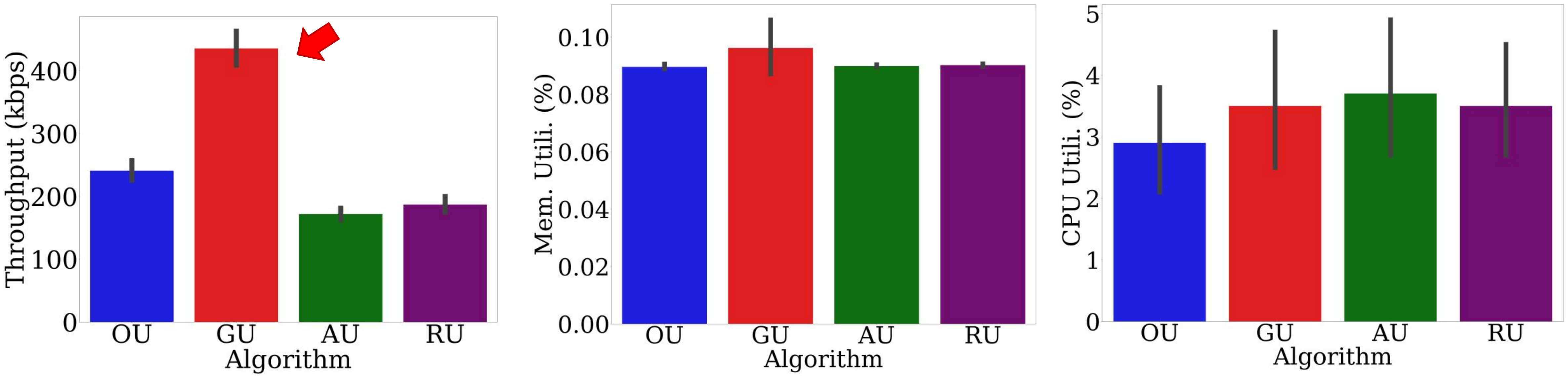




- Our OU and GU algorithms reduce expected and actual state deviation in both optimization and real testbed

Overhead of Algorithms

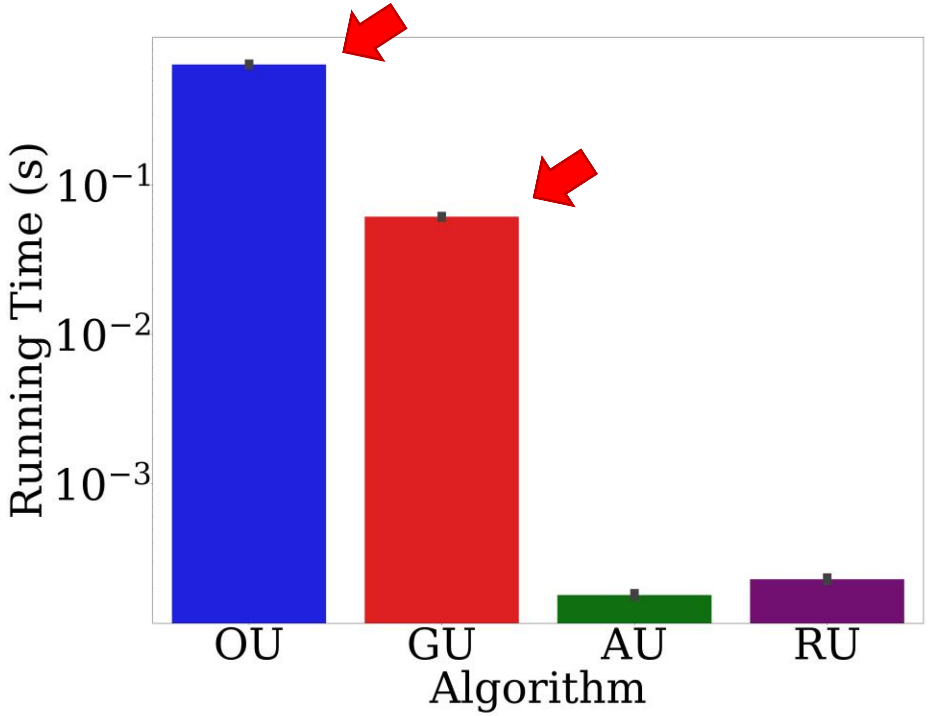
- GU only consumes ~ 400 kbps of total bandwidth, which is not high in modern networks
- State synchronization algorithms impose no impact on the application server overhead



Running Time of Algorithms

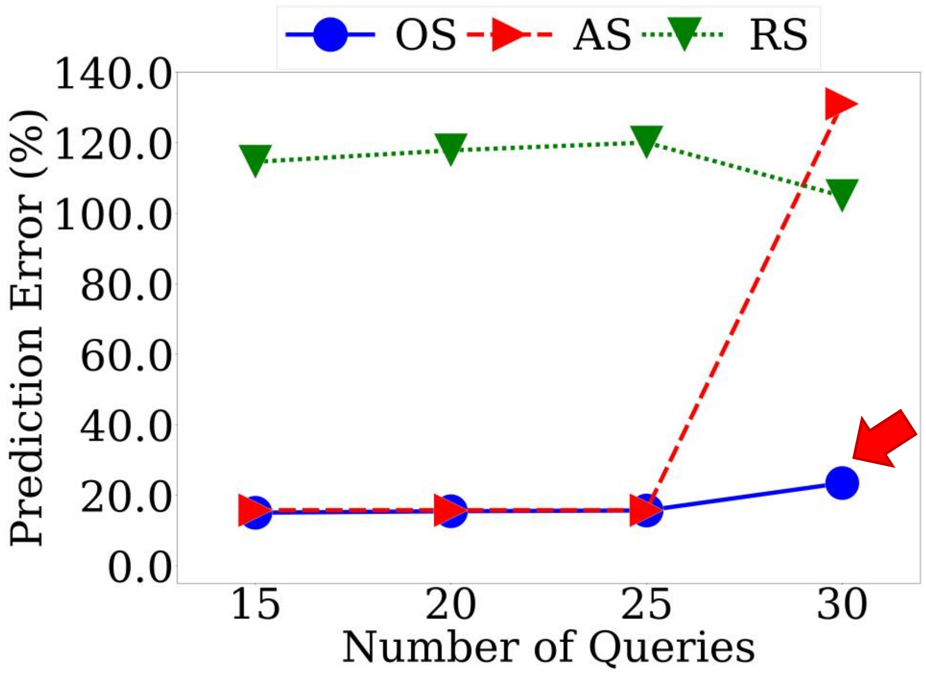
- GU runs much faster than the OU , with a 10.46 times speed-up in terms of the running time
- The running time of OU is merely ~ 800 ms

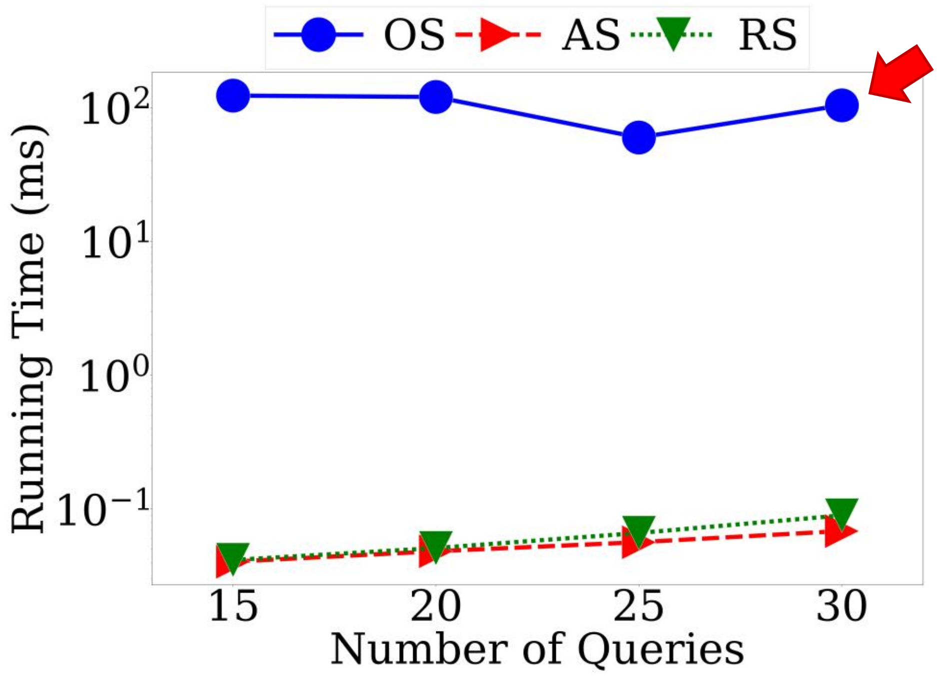
- The overhead of OU and GU algorithms is acceptable



Performance of different what-if analysis algorithms with varying numbers of queries

- OS leads to the lowest prediction error: up to 107.55% and 103.35% reduction
- OS incurs almost constant running time, up to 101.59 ms

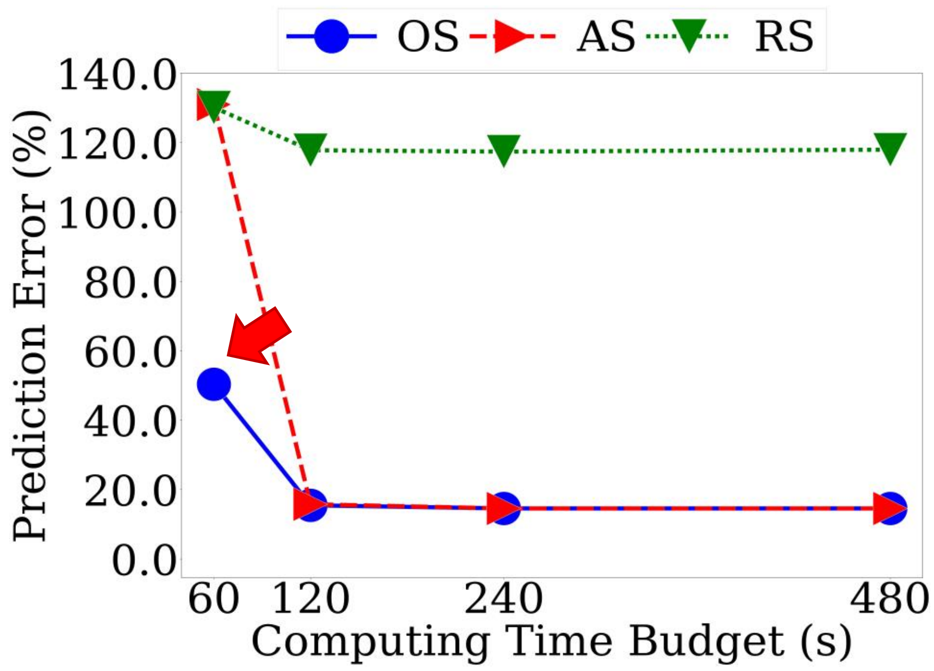


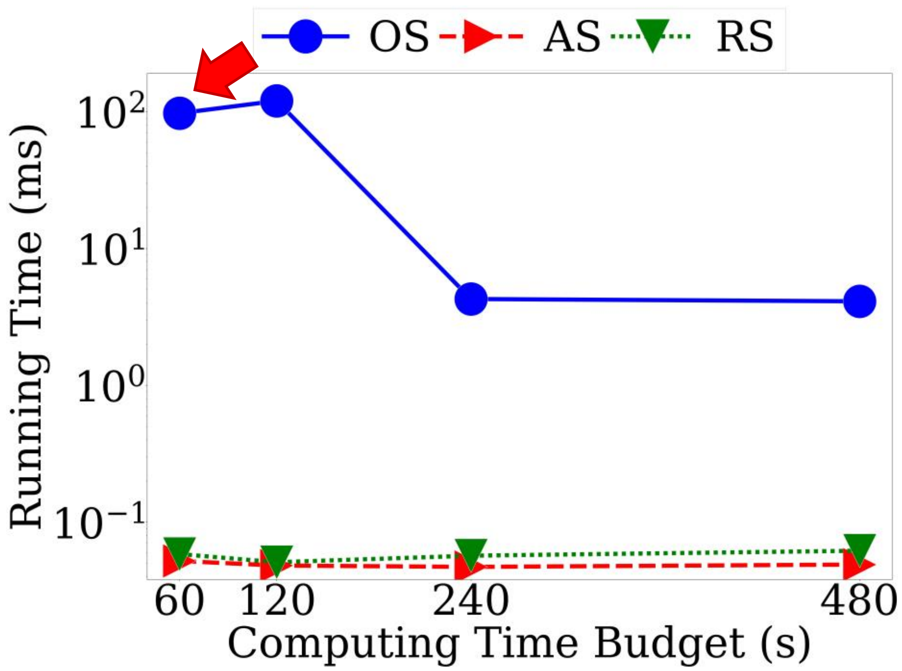


Evaluations

Performance of different what-if analysis algorithms with varying computing time budgets

- OS achieves up to 75.77% and 83.63% lower prediction error than AS and RS, and completes within 70.67 ms under a 60-s computing budget

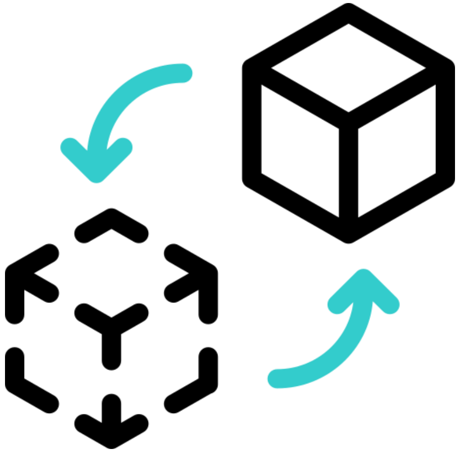




- Our OS algorithm delivers the smallest prediction error and completes within 1 second

Outline

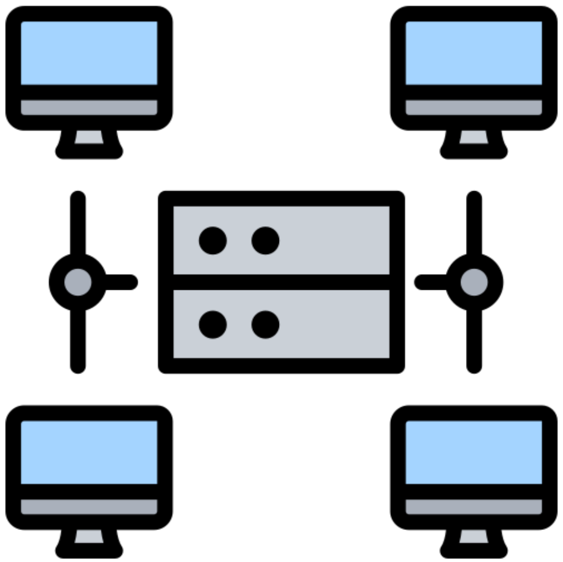
- Introduction
- Goal and Challenges
- Related Work
- System Overview
- Problems and Solutions
- Implementations
- Evaluations
- Conclusion & Future Work



Conclusion

- ① Developed and optimized an NDTC for IoT-instrumented smart environments
- ② Proposed OU and GU algorithms to minimize state deviation between PTs and DTs by optimizing update frequency and data granularity
- ③ Proposed the OS algorithm to select the best what-if analyzer, reducing QoS prediction errors within the computing time budget

Future Work



① Network topology

② Human-in-the-loop

SUMO

③ Domain-specific simulators

Thank you for listening

Thanks for the help of Prof. Hsu, Cheng-Chia Lai, and all labmates

Q&A

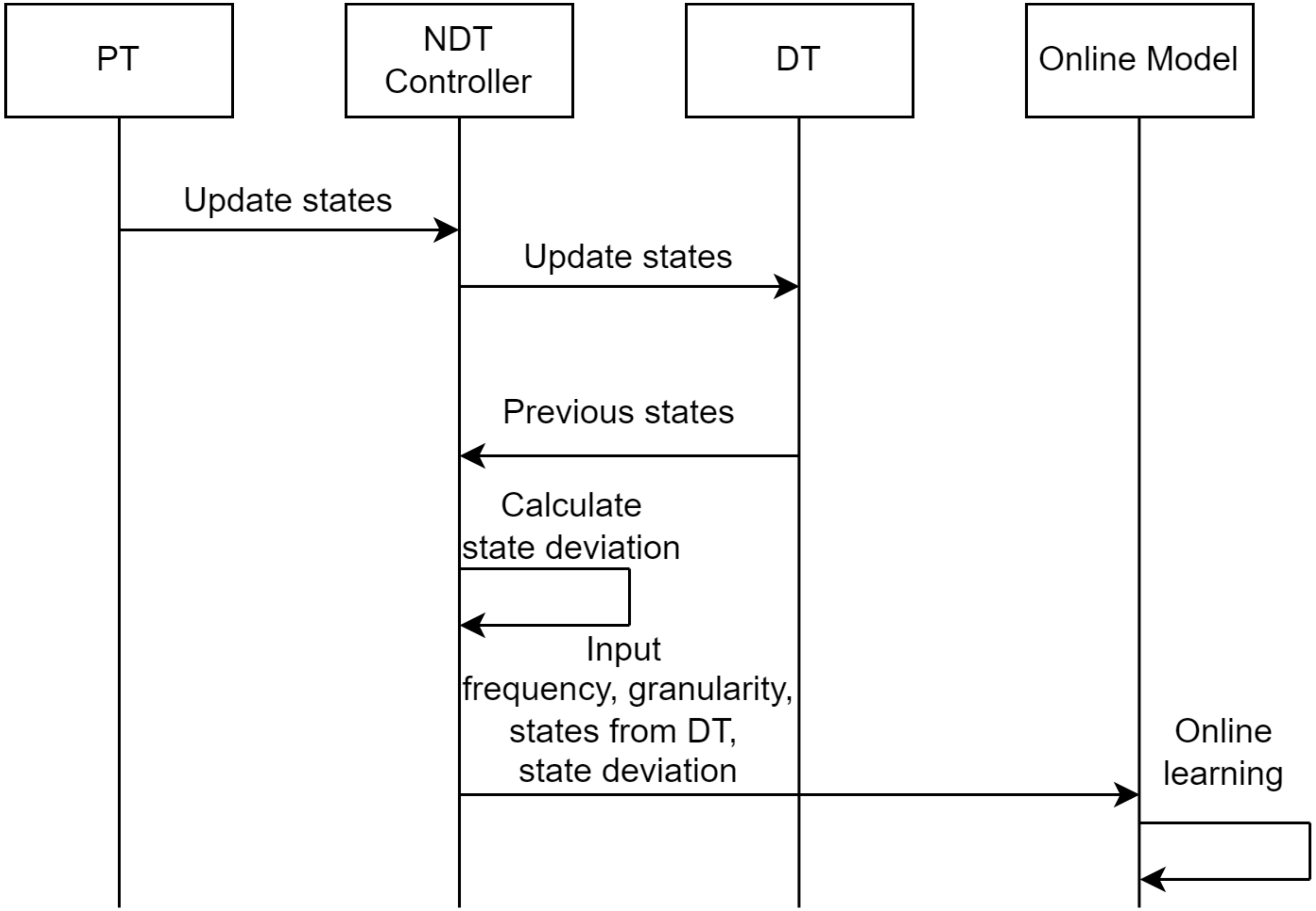
Publications:

- G. Shi, **C. Wu**, and C. Hsu. Error Concealment of Dynamic LiDAR Point Clouds for Connected and Autonomous Vehicles, in Proc. of IEEE Global Communications Conference (GLOBECOM'23), Kuala Lumpur, Malaya, December 2023
- **C. Wu**, R. Rahman, C. Hsu, C. Lai, N. Venkatasubramanian, and C. Hsu. Evaluating Subsurface Single-Hop WiFi and LoRa Networks for Internet of Underground Things, in Proc. of IEEE Global Communications WorkShop (GLOBECOM'23), Kuala Lumpur, Malaya, December 2023
- T. Chang, **C. Wu**, G. Bouloukakis, C. Hsu, and N. Venkatasubramanian. SmartParcels: Constructing Smart Communities Through Human-in-the-Loop Urban IoT Planning, ACM Transactions on Internet of Things, 2024. (**Under Review**)
- **C. Wu**, C. Lai and C. Hsu. Optimizing Network Digital-Twin Controllers for Internet-of-Things Instrumented Smart Environments, in Proc. of IEEE International Conference on Cloud Computing Technology and Science (CloudCom'24), Abu Dhabi , UAE, December 2024. (**Under Review**)

PT/DT' s States

- Network states:
  - flow state, port state
  - network interface states (TX/RX packets, bytes, errors dropped, overruns, carrier, collisions)
- Hardware states: CPU utilization, memory utilization
- IoT states: sensor data, timestamp

Workflow of NDTC



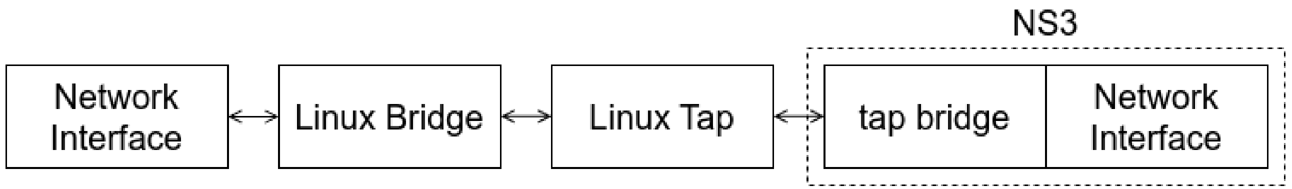
Network Emulator

- Build a Linux bridge and a virtual network interface **TAP** (Ethernet tunnel software network interface)

- Bind the node in the ns3 simulation to the TAP

- Receive RTMP streaming from a workstation and retransmit the packets to the TAP

- **Simulate the real traffic flow from the testbed in the simulation**



https://www.nsnam.org/wiki/HOWTO_Use_Linux_Containers_to_set_up_virtual_networks