Oceanus: Scheduling Traffic Flows to Achieve Cost-Efficiency under Uncertainties in Large-Scale Edge CDNs

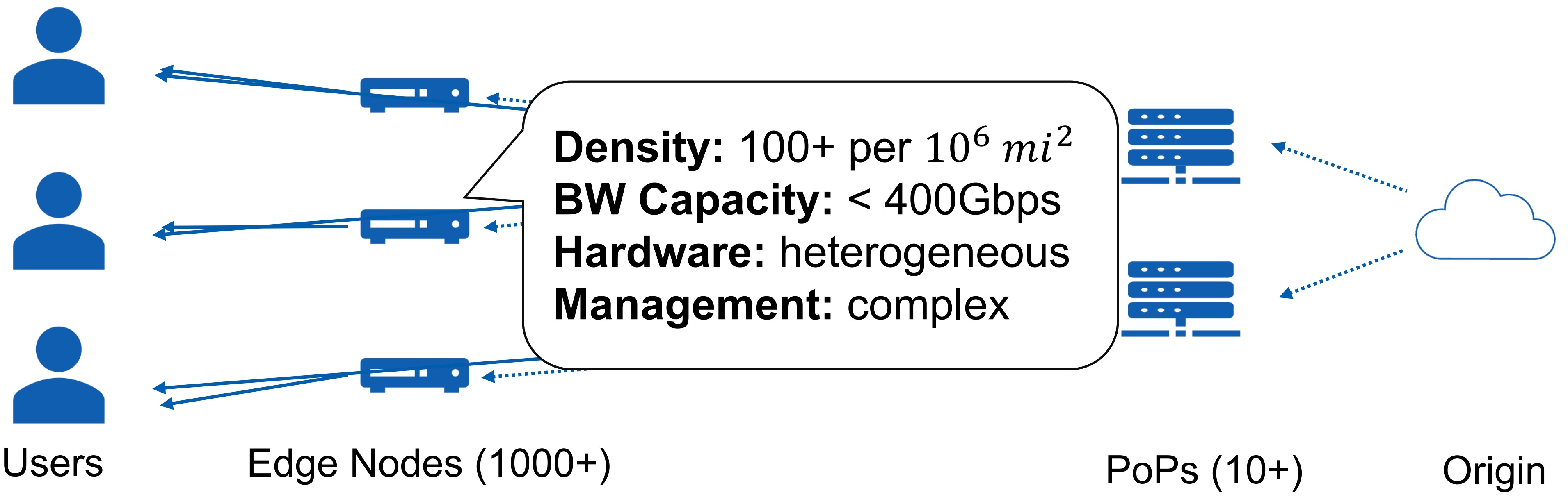
Chuanqing Lin, Gerui Lv, Fuhua Zeng, Hanlin Yang, Junwei Li, Xiaodong Li,

Jingyu Yang, Yu Tian, Qinghua Wu, Zhenyu Li, and Gaogang Xie

University of Chinese Academy of Sciences (UCAS)
Alibaba Cloud
Purple Mountain Laboratories
Computer Network Information Center, Chinese Academy of Sciences (CNIC, CAS)

Background: Large-Scale Edge CDNs

- Content Delivery Networks (CDNs) evolve to *edge CDNs*
- Trade-off:
  - Pros: lower access latency
  - Cons: more vulnerable, higher management complexity and operating cost.



Edge CDNs require fine-grained and careful traffic scheduling.

Edge CDN Scheduling

- Goals for scheduling:
  1. Primary: guaranteeing **performance SLA** (Constraints)
  2. Secondary: minimizing **bandwidth costs** (Objective)
- Sequential workflow: **Bandwidth planning → Flow scheduling**
  - Jointly optimization incorporates unacceptable scales (Timeslots * Nodes * Flows ~ $\mathcal{O}(10^{12})$)

Bandwidth Planning

Goal: minimizing global billable bandwidth
Scale: Timeslots (~8640) * Nodes (~2000)
Complexity: NP-hard (MILP)

→ Bandwidth Budget

Flow Scheduling

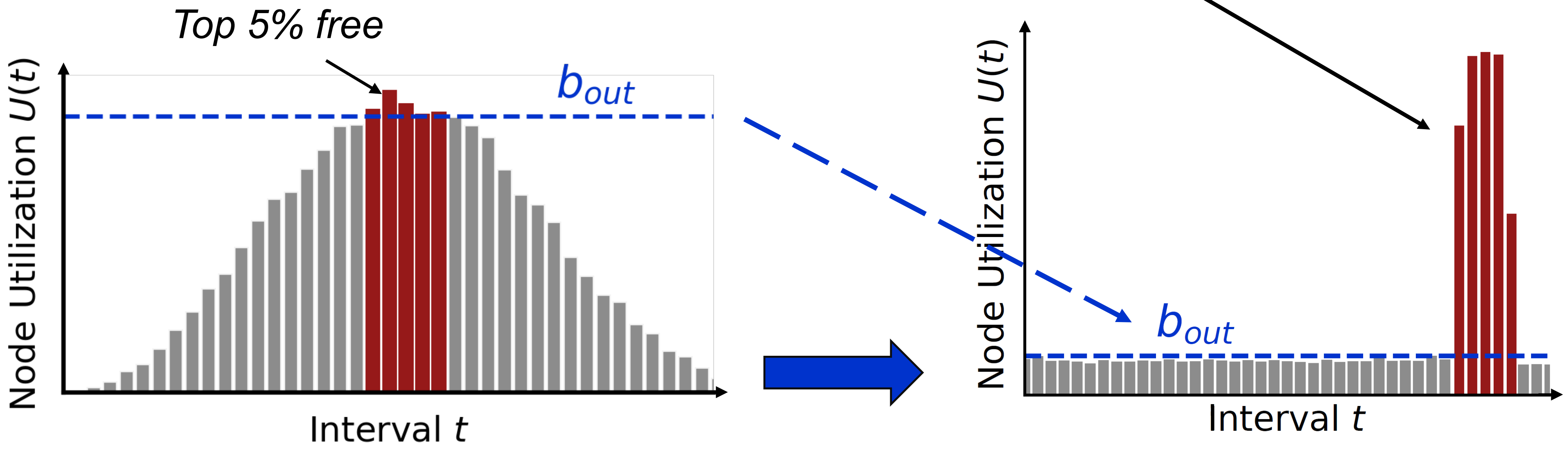
Goal: guaranteeing performance SLAs
Scale: Flows (~10000) * Nodes (~2000)
Complexity: NP-hard (MILP)

Cost Model: 95th-percentile Billing

- Cost = rate * Billable Bandwidth

$$b_{out} = P_{95}\big(U_{out}(t)\big)$$

- Traffic engineering:
  - **How much:** *Minimizing* billable bandwidth $\sum b_{out}$;
  - **Which and When:** *Maximizing* the utilization of free augmenting slots.

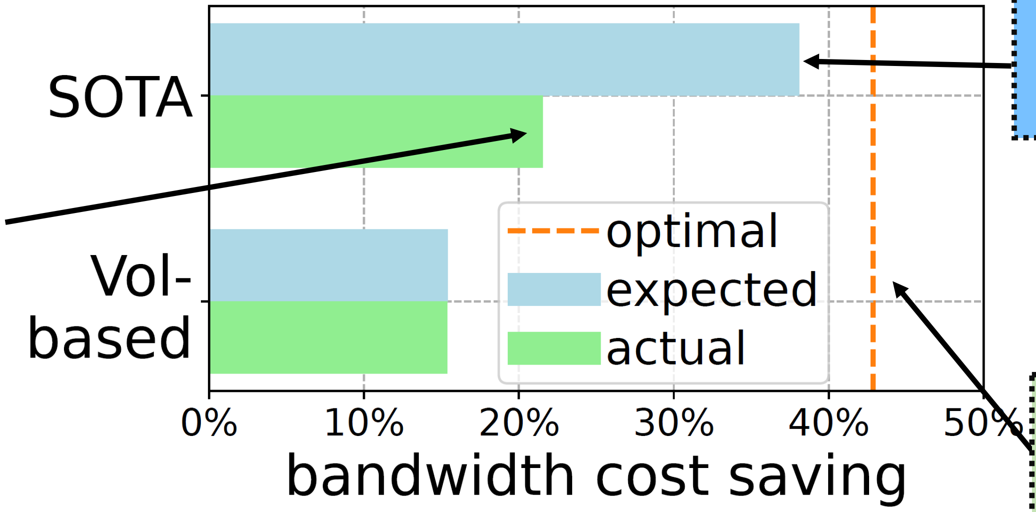


Problem: Cost-saving Gaps from Optimal to Reality

- **Issue:** Current approaches fail to achieve their claimed bandwidth cost-saving in production.
- Evaluation Result: $Cost_{actual} \gg Cost_{expected} \gg Cost_{optimal}$

SOTA claims to save 38% bandwidth cost.

SOTA only achieves 21% bandwidth cost eventually.

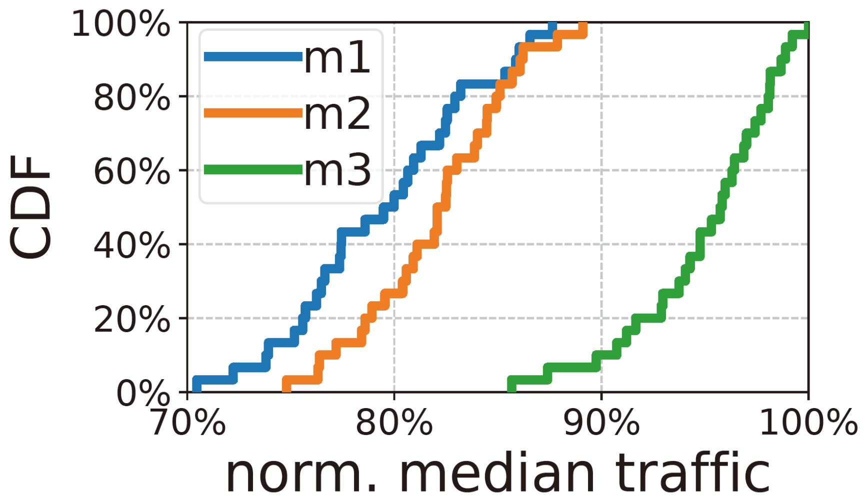


At most 42% bandwidth cost can be saved theoretically.

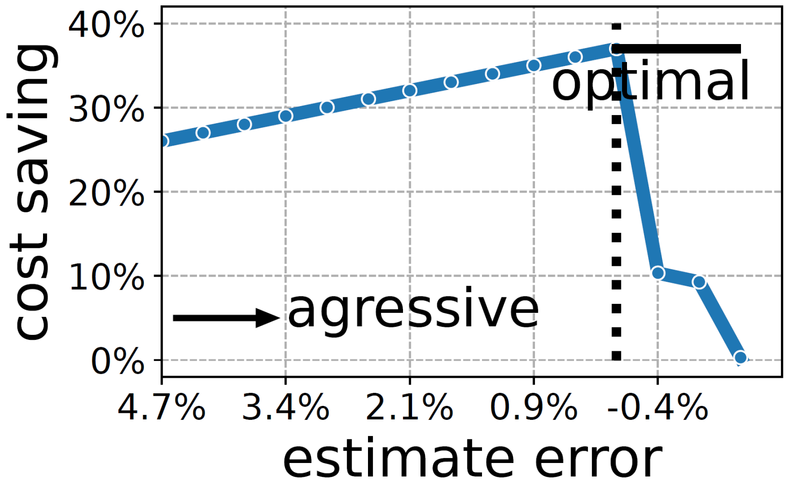
Root cause: Overlooked uncertainties in real-world lead to node utilization discrepancies.

Uncertainty #1: Traffic Demand Dynamics

- **Paradigm:** Bandwidth budgets computation is based on historical traffic demands.
- **Issue:** Traffic demands vary widely every month.
- **Consequence:** Underestimated traffic demands severely degrade cost-saving.
- Strawman solution: Frequently update bandwidth budgets.



Daily median traffic demand over 3 months.

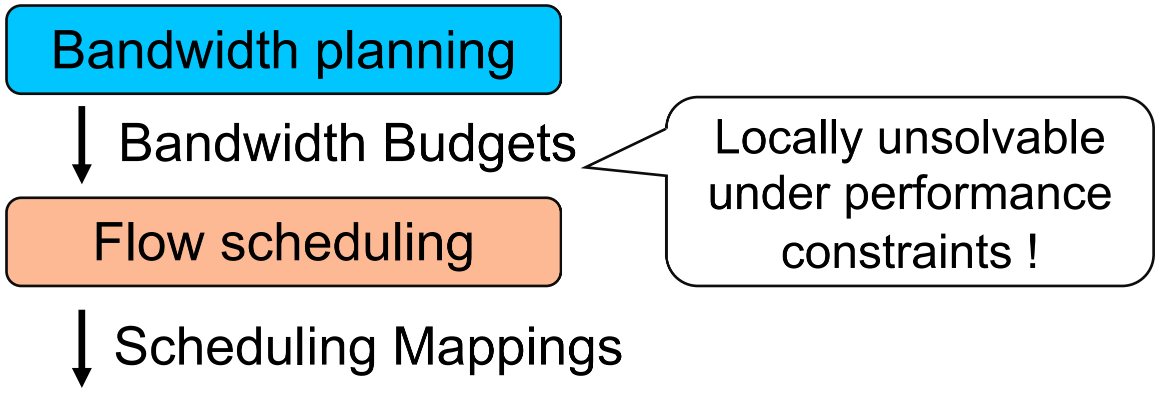


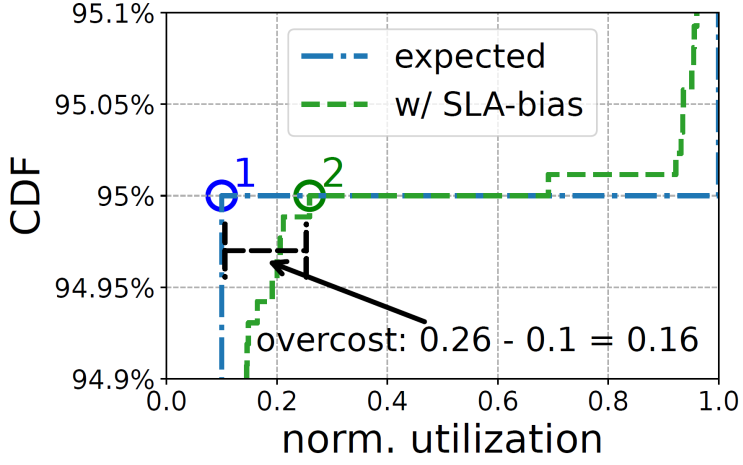
Underestimated traffic demand leads to severe cost-saving degradation.

Challenge: Strategy update is NP-hard and computationally expensive.

Uncertainty #2: SLA-Constrained Scheduling Bias

- **Paradigm:** Bandwidth planner does not consider performance assurance requirements in sequential workflow.
- **Issue:** Bandwidth budget can be locally suboptimal.
- **Consequence:** Bandwidth budget is intentionally violated for SLA assurance.
- Strawman solution: jointly consider performance in bandwidth planner.



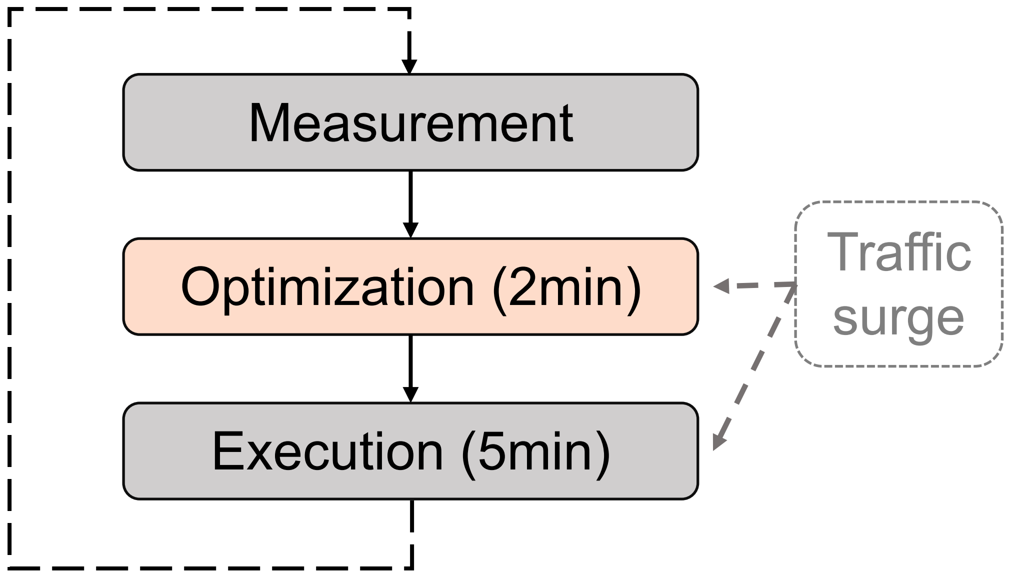


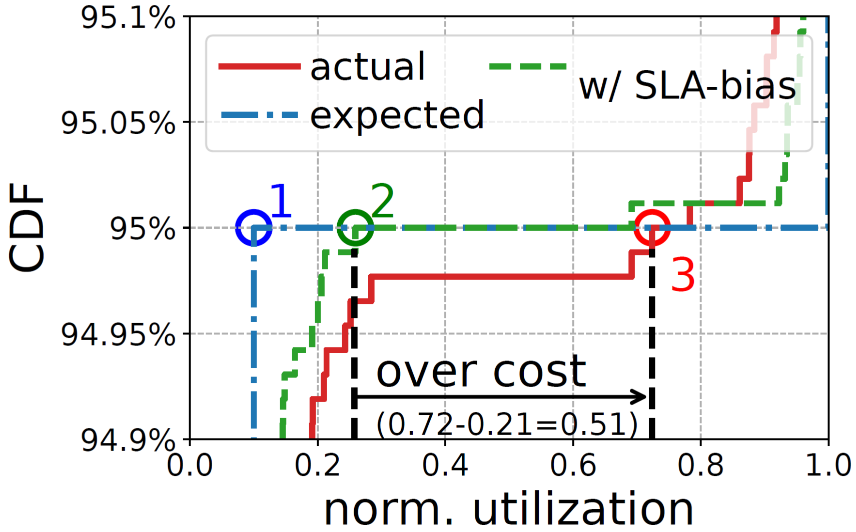
Node over-utilizations incur additional costs.

Challenge: Jointly optimization for cost and performance is computationally unacceptable.

Uncertainty #3: Systemic Scheduling Deviation

- **Paradigm:** Bandwidth budgets are made based on the latest measurement.
- **Issue:** Strategy execution delay and traffic surge exist.
- **Consequence:** Actual utilization ≠ Expected utilization
- Strawman solution: Consider node-overutilization as augmentation.





Node over-utilizations incur additional costs.

Challenge: Inevitable systemic deviation depletes augmentation slots prematurely.

Solution

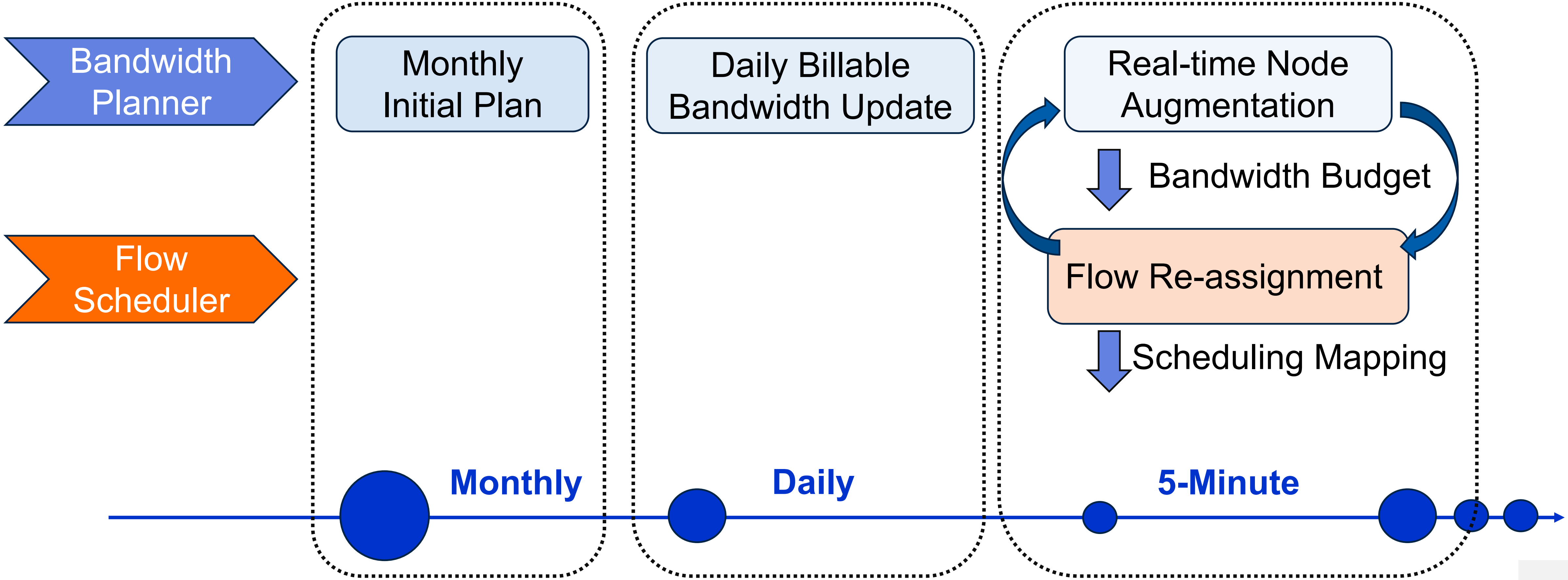
- Goal: achieving cost-efficiency under uncertainties
- Key idea:
  - Proactively reduce future uncertainties;
  - Reactively adapt to existing uncertainties.

Uncertainty 1 Traffic demand dynamics	**Uncertainty 2** SLA-constrained bias	**Uncertainty 3** Systemic deviation
Challenge 1 How to reconcile long-term planning with real-time adjustment?	**Challenge 2** How to coordinate separated components under performance constraints?	**Challenge 3** How to mitigate the impact of node utilization discrepancies with limited resources?
Solution 1 Decoupling bandwidth planning to operate on multiple timescales	**Solution 2** Coordinating two components with bidirectional feedback	**Solution 3** Augmenting nodes with minimal augmentation marginal cost

Oceanus: Coordinated traffic scheduling on multiple timescales

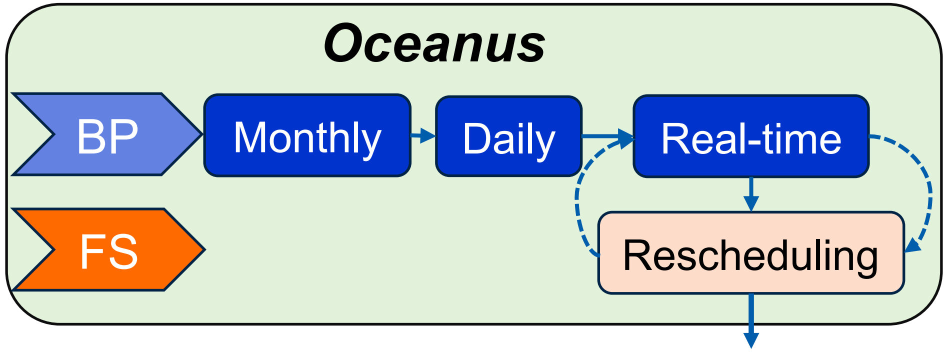
Oceanus Design Overview

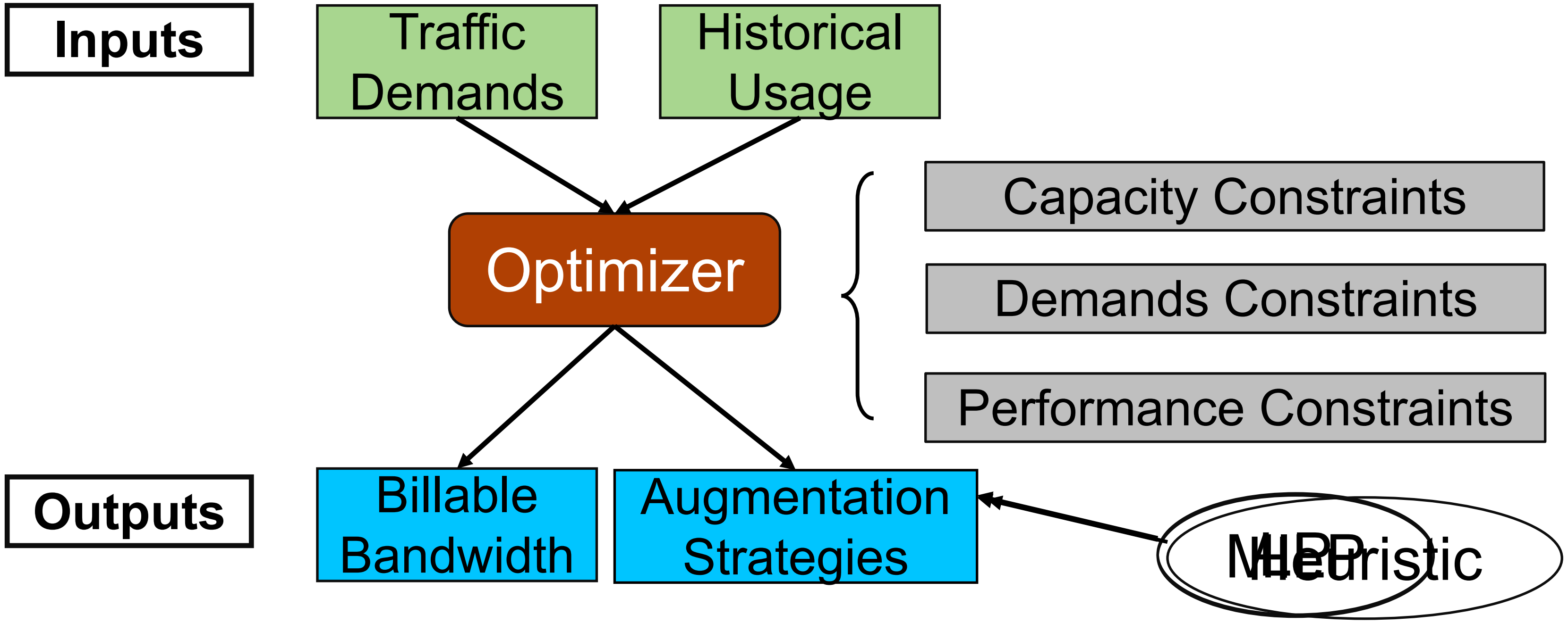
- Bandwidth Planner: updating **Bandwidth Budgets** for nodes
- Flow Scheduler: updating **Scheduling Mappings** for traffic flows

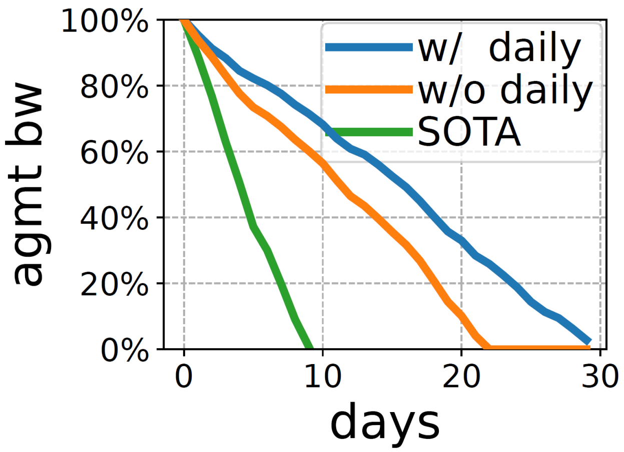


Strategy 1: Multi-Timescale Planning

- **Goal:** *adapting* to traffic demand dynamics
- Month start: initialize solution
- Daily: track and adapt to long-term dynamics
- 5-Minute: augment nodes for traffic surge



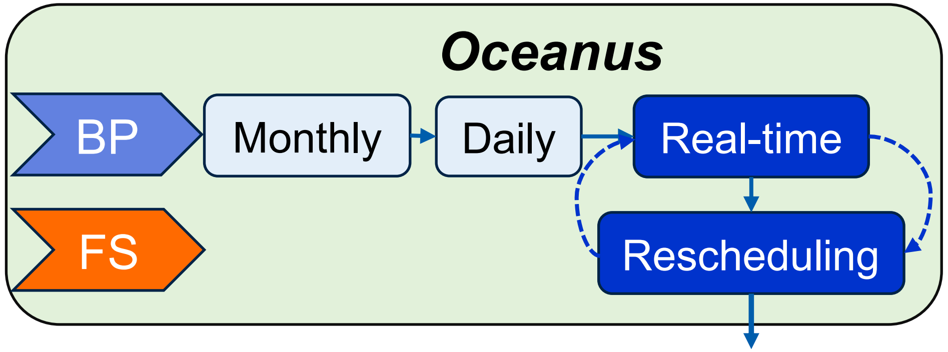


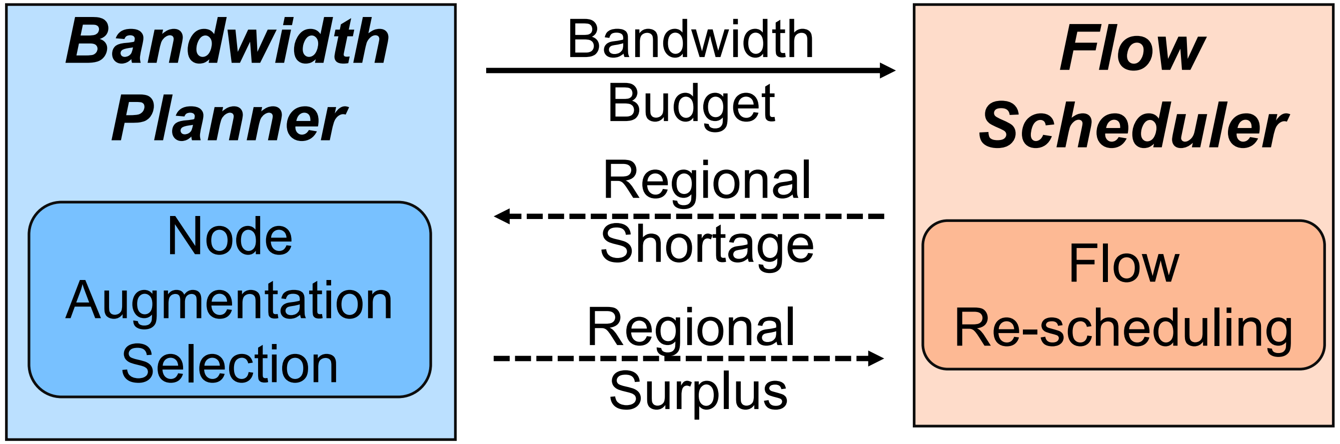


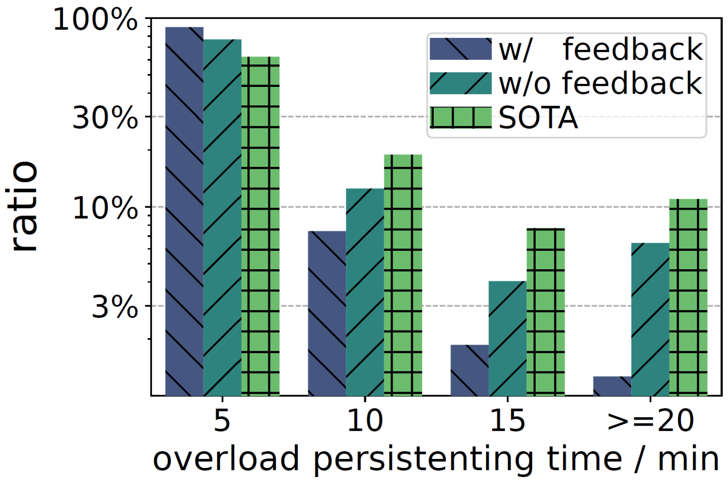
Consumption rate of augmentation resources

Strategy 2: Bidirectional Feedback

- **Goal:** *reducing* SLA-Constrained bias
- Coordinated BP and FS
  - FS: report regional resource shortage
  - BP: suggest resource surplus region

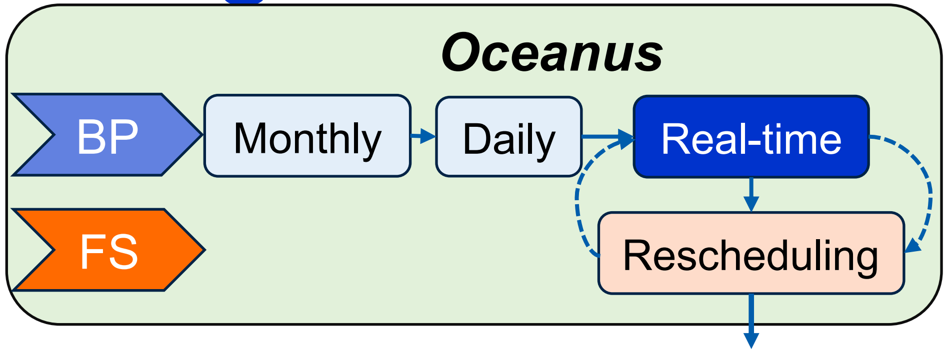




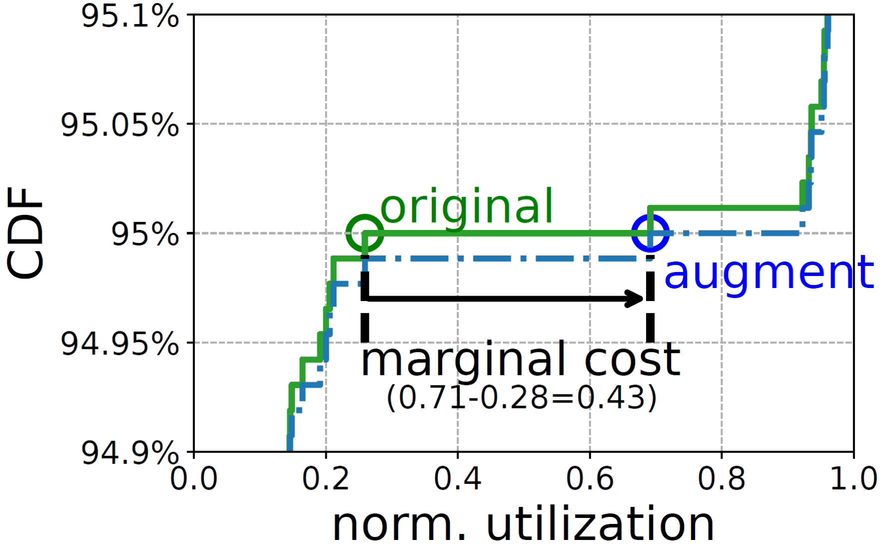


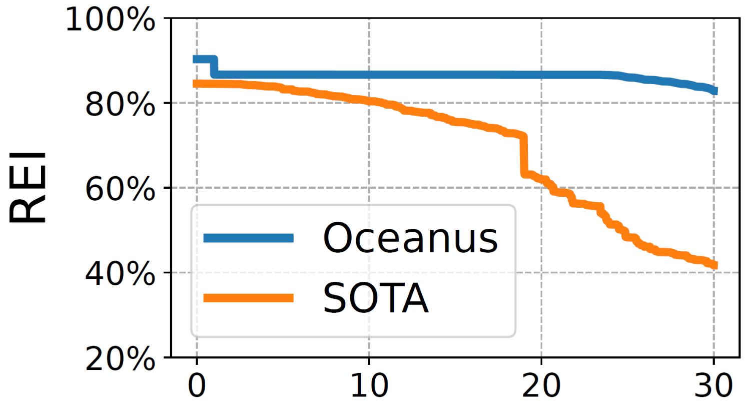
89.4% of over-utilization events are solved within 5 minutes.

Strategy 3: Marginal Cost-Based Augmentation



- **Goal:** *adapting* to systemic deviation
- Prioritize node with the minimal augmentation **marginal cost** to augment
    - Marginal cost = *Price*(augment in next slot) – *Price*(not augment)





Oceanus limits the cost-saving degradation when augmentation resources are exhausted.

Evaluation Setup

- **Testbed setup:** replaying real-world collected traffic
  - **Dataset:** 3 months, **Tbps+** real-time traffic in **Alibaba Cloud CDN**, served by **2,300+ nodes**
  - **Scenario:** escalating realism
  - **Baselines:**
    - Percentile-based: **CASCARA** (SOTA)
    - Volume-based: **ENTACT**
  - **Metrics:**
    - Relative Exploitation Index *(REI)* → Cost-saving potential exploitation, higher is better
    - Percentile Billing Ratio *(PBR)* → Final billed percentage, lower is better
- **Real World deployment:** evolving flow scheduler in production
  - Baseline: Global Flow Scheduler
  - Metrics: RTT/ Update speed/ Mapping stability

Scenario 1:
Traffic demand dynamics

↓

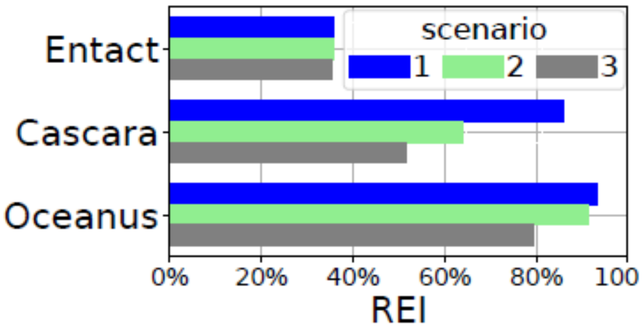
Scenario 2:
Scenario 1
+ SLA-constrained bias

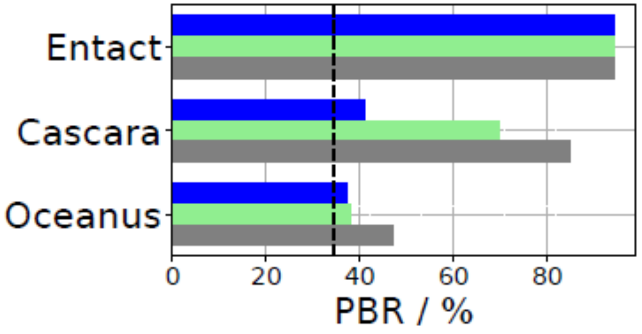
↓

Scenario 3:
Scenario 2
+ Systemic deviation

Key Results

- Oceanus **saves the most bandwidth costs**
  - Average cost saving: **2.6**× vs. CASCARA
- Oceanus **maintains cost-saving performance** under uncertainties
  - Oceanus achieves **79.4%** of the theoretical maximum cost-savings
- Oceanus **updates** scheduling mappings **faster and more stably**
  - vs. Global Scheduler: computation time **40.3%**▼, mapping change ratio **78.3%** ▼





	Local scheduler		Global scheduler
Over-utilized nodes	**4.0**	←	5.7
RTT / ms	**15.3**	←	16.1
Mapping change ratio	**2.5%**	←	11.5%
Computation time / s	**127.3**	←	213.1

Oceanus successfully achieved its design goal: achieving cost-efficiency under uncertainties.

Summary

- **Observation:** We identified that real-world uncertainties in large-scale edge CDNs create a fundamental mismatch between bandwidth cost planning and execution.

- **Solution:** We designed Oceanus, a multi-timescale coordinated traffic scheduling system for optimizing bandwidth costs and performance on large-scale edge CDNs.

- **Impact:** Oceanus is effective in achieving **21%** cost-savings under uncertainties on Alibaba Cloud CDN with Tbps+ real-time traffic.

Thanks!

Q & A

For any further questions, please contact:

Chuanqing Lin (UCAS)

linchuanqing20b@ict.ac.cn