EA-Market: Empowering Real-Time Big Data Applications with Short-Term Edge SLA Leases

Ruozhou Yu, ***North Carolina State University***

Huayue Gu, Xiaojian Wang, Fangtong Zhou, *North Carolina State University*

Guoliang Xue, *Arizona State University*

Dejun Yang, *Colorado School of Mines*

NC STATE UNIVERSITY

Outlines

Background and Motivation

System Modeling

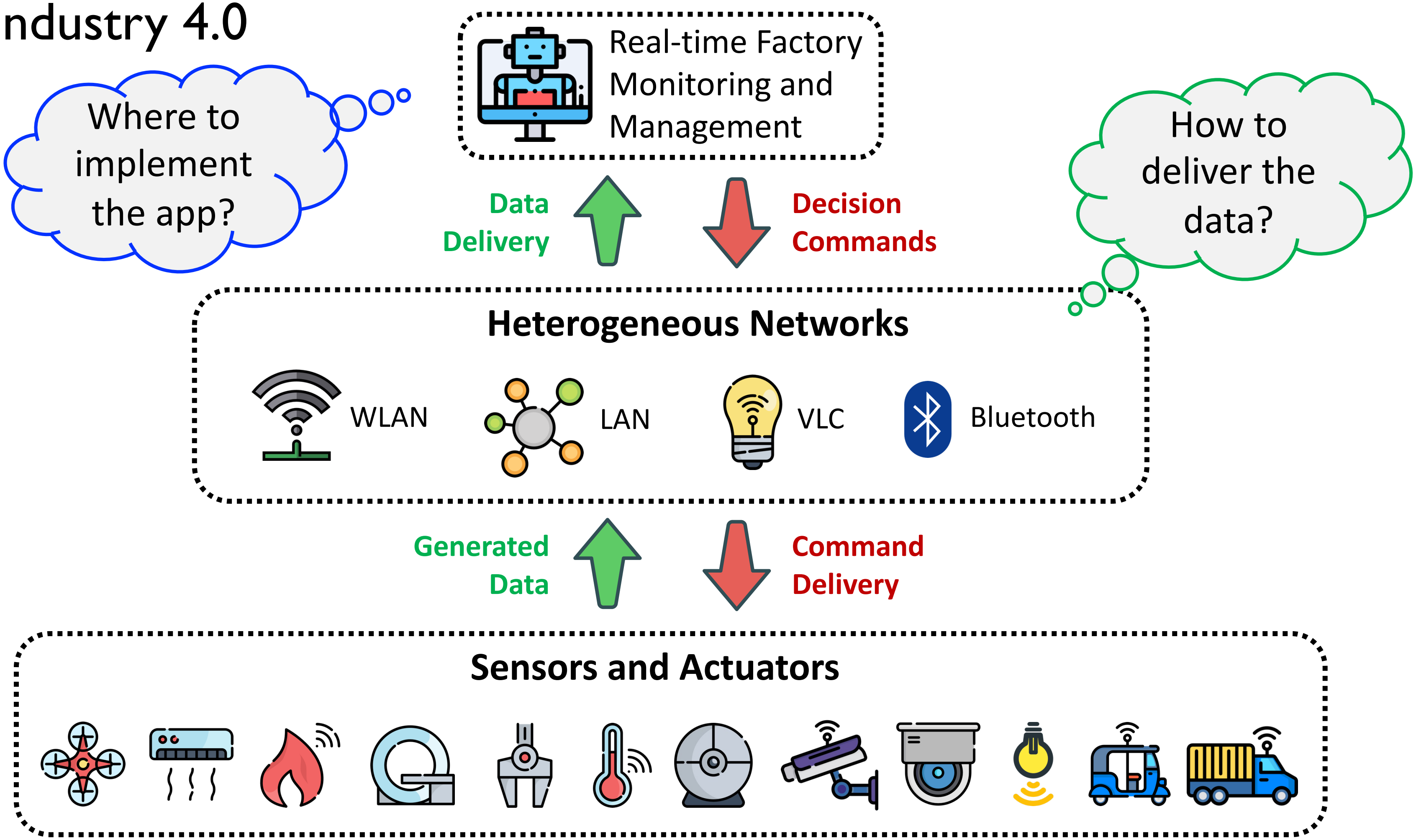
Solution Design

Performance Evaluation

Discussions, Future Work and Conclusions

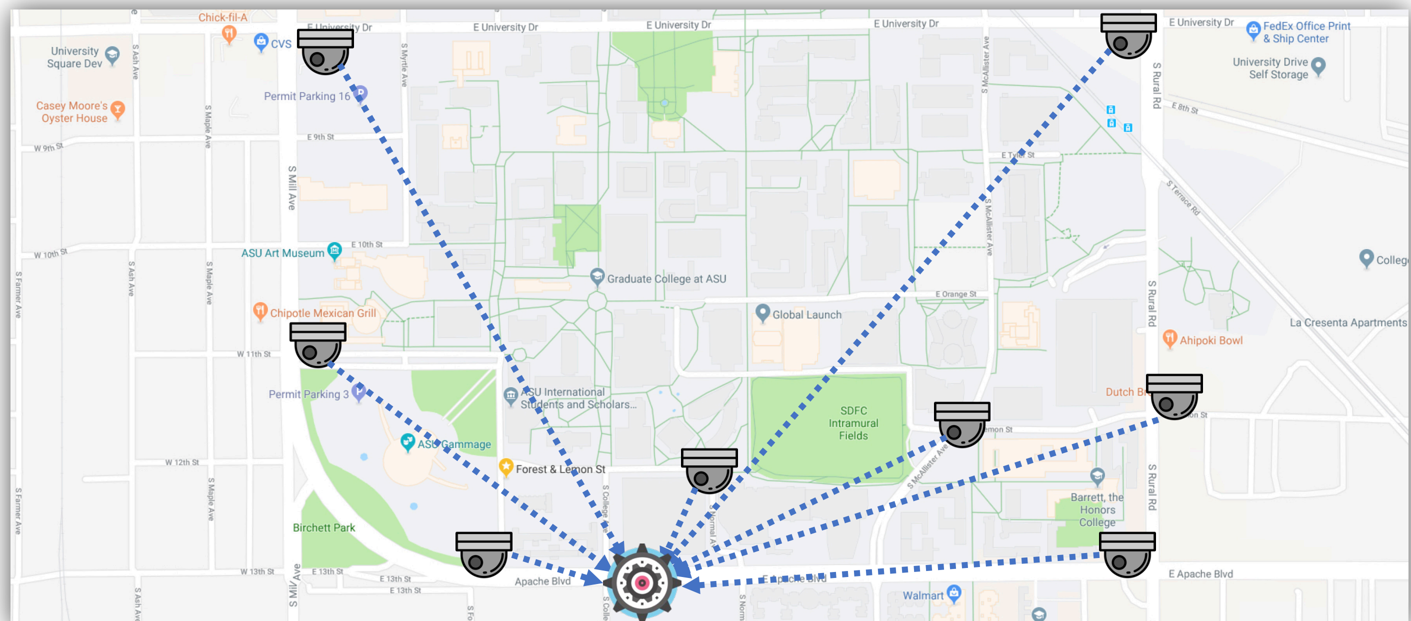
A Typical Scenario in Edge

- Industry 4.0



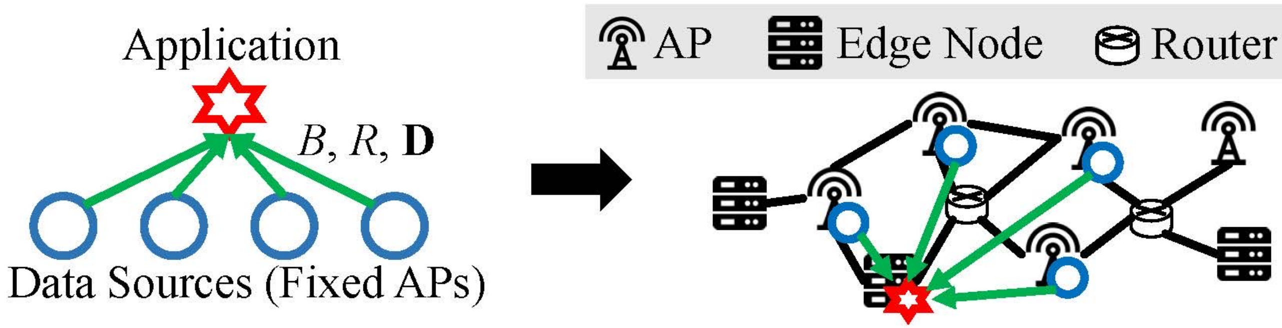
Real-time IoT Applications

- Application = Logic + Data
  - Logic: data processing unit
  - Data: from multiple sources (sensors) in the network

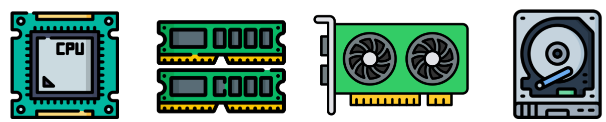


- ***Requirements:***
  1) Bandwidth and resource: transmit and process with enough speed
  2) Real-time: channel latency up to a required bound

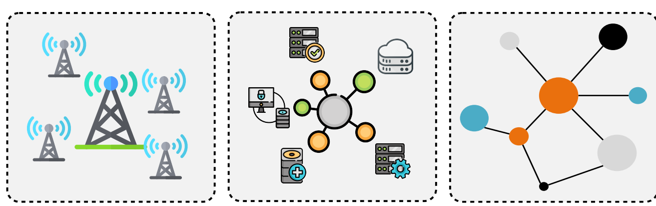
Resource and Performance Requirements



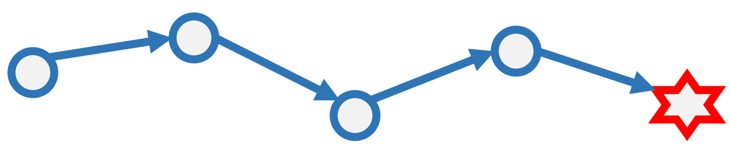
- Computing resources:



- Network bandwidth:

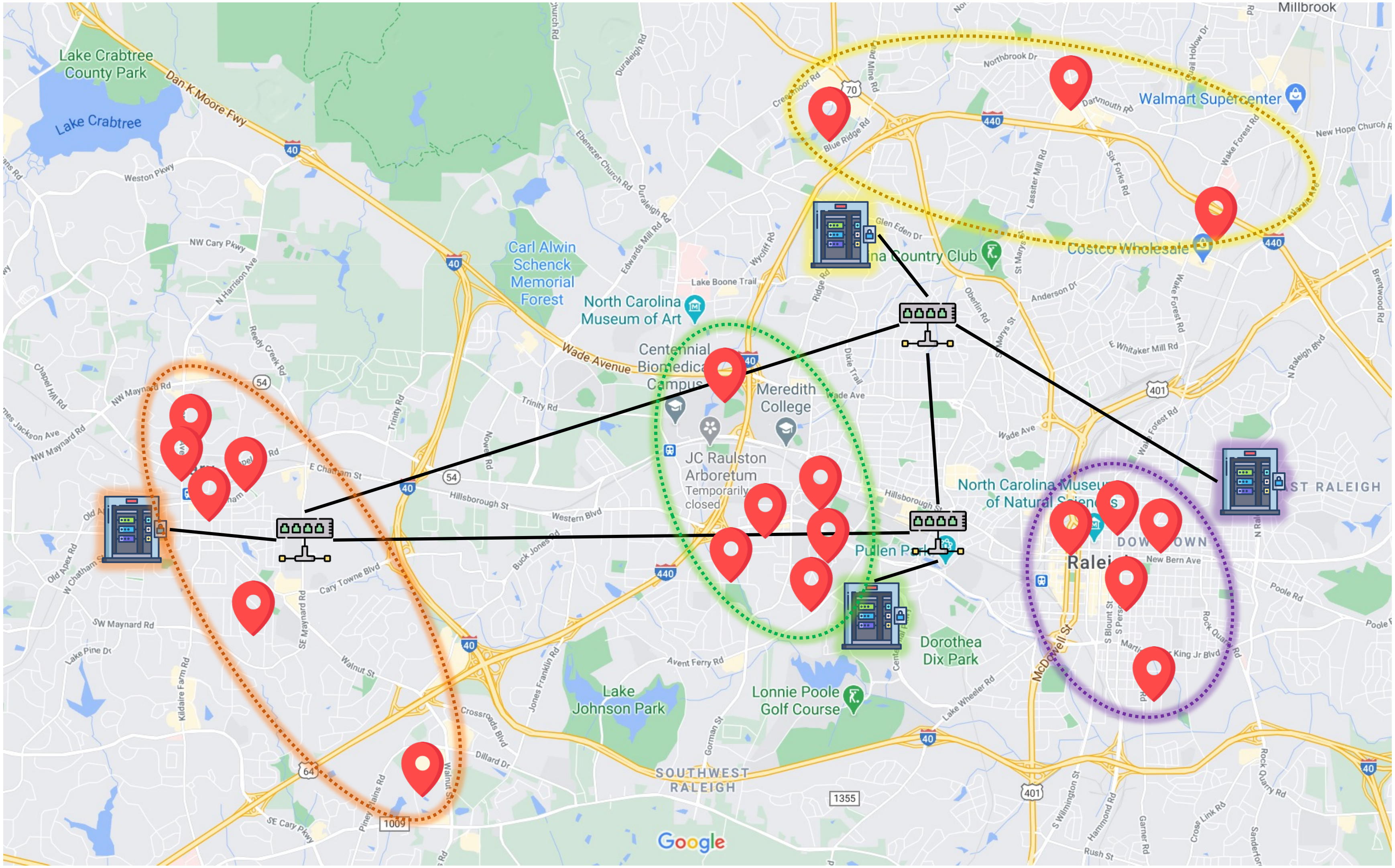


- End-to-end latency:



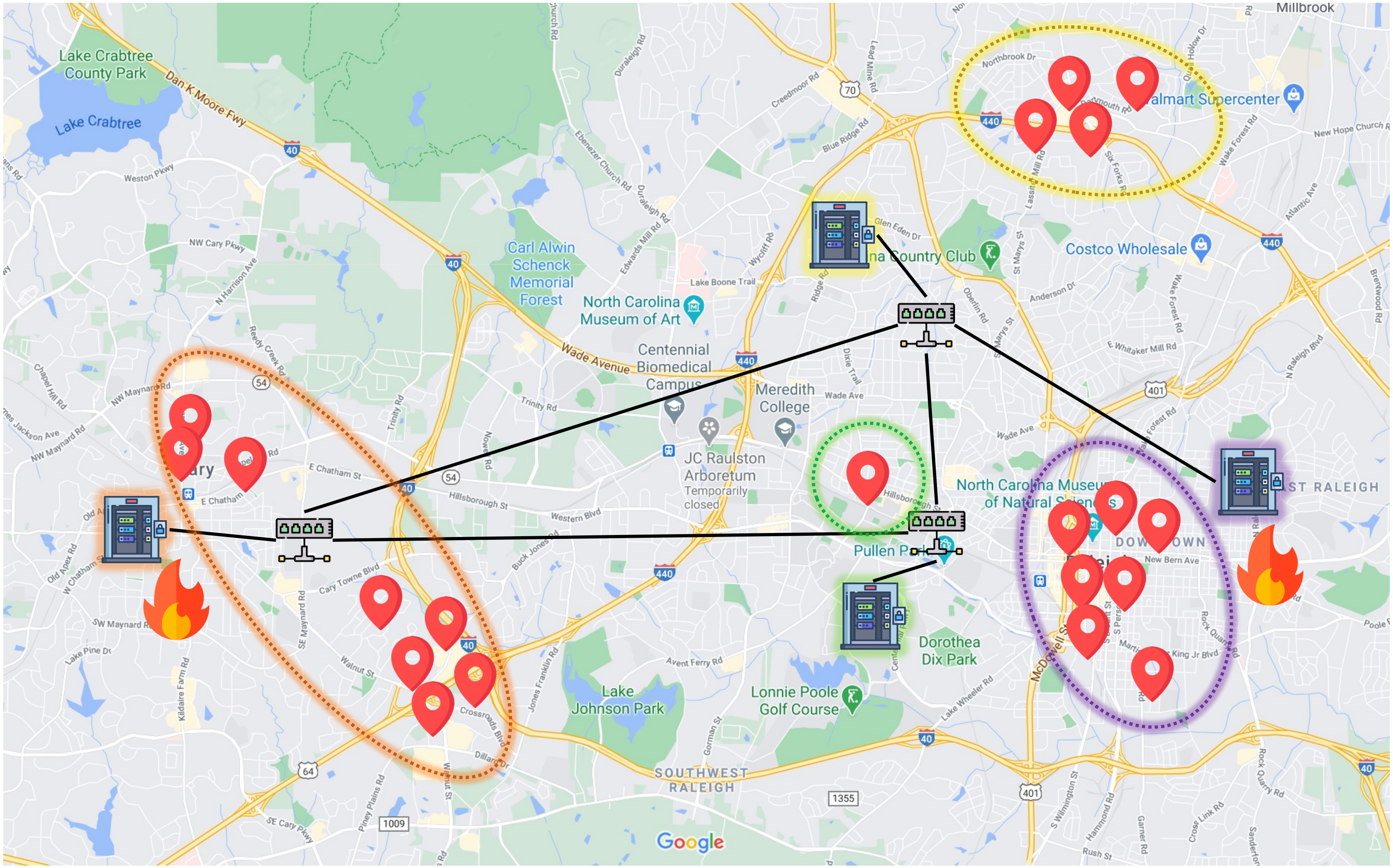
Service-Level Agreement (SLA) Guarantees

Geo-Distributed Services & Edge Computing



NC STATE UNIVERSITY

Time-Varying Demands in Geo-Distributed Apps



The SLA Dilemma and Short-term Edge SLAs

Without edge SLA
- Users served in best effort
- Performance degradation when congested
- No priority among application traffic and demands

With long-term edge SLA as cloud
- SLAs must be provisioned for peak demands of applications
- High SLA price for applications
- Wasted resources at idle times
- Violation due to demand variation

Our solution: short-term edge SLAs
- ✓ Applications **dynamically request** SLAs based on predicted demand
- ✓ Edge provider **dynamically provisions** resources to fulfill SLAs
- ✓ Pricing negotiated based on instantaneous **demand and supply**
- ✓ Financial-driven **prioritization**

Question

How to design an efficient and strategy-proof short-term SLA market mechanism?

Methodology Overview

Inputs:

App Structure & Demands
Async., Sync., Centralized

Edge Network
Topology, geo-dist. sources

Market Mechanism:

Online Application SLA Provisioning
- Short-term SLA leases
- Random SLA request arrivals
- Resource, bandwidth, delay

Service Admission and Pricing
- Admission based on demand and supply
- Pricing based on current utilization

Outputs:

Supply and Demand-driven SLA Provisioning
Competitive social welfare, truthfulness, rationality, efficiency

Outlines

Background and Motivation

System Modeling

Solution Design

Performance Evaluation

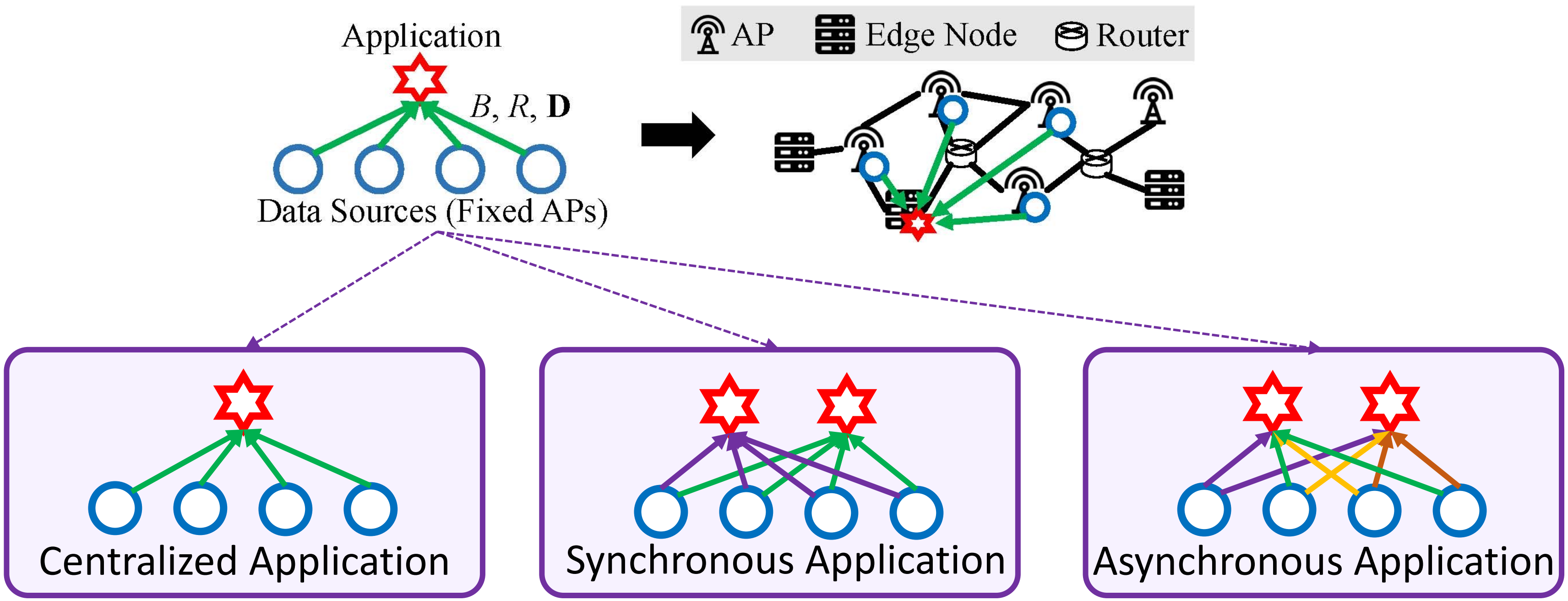
Discussions, Future Work and Conclusions

System Model: Application SLA Provisioning

Given: Edge network + edge nodes | Edge Apps

Find: App-node mapping | Routes + Bw Alloc | Resource Allocation | Pricing

S.T.:
- Routing delay <= app QoS requirement
- Max link/node load <= provider threshold



Problem Statement: Online Market Design

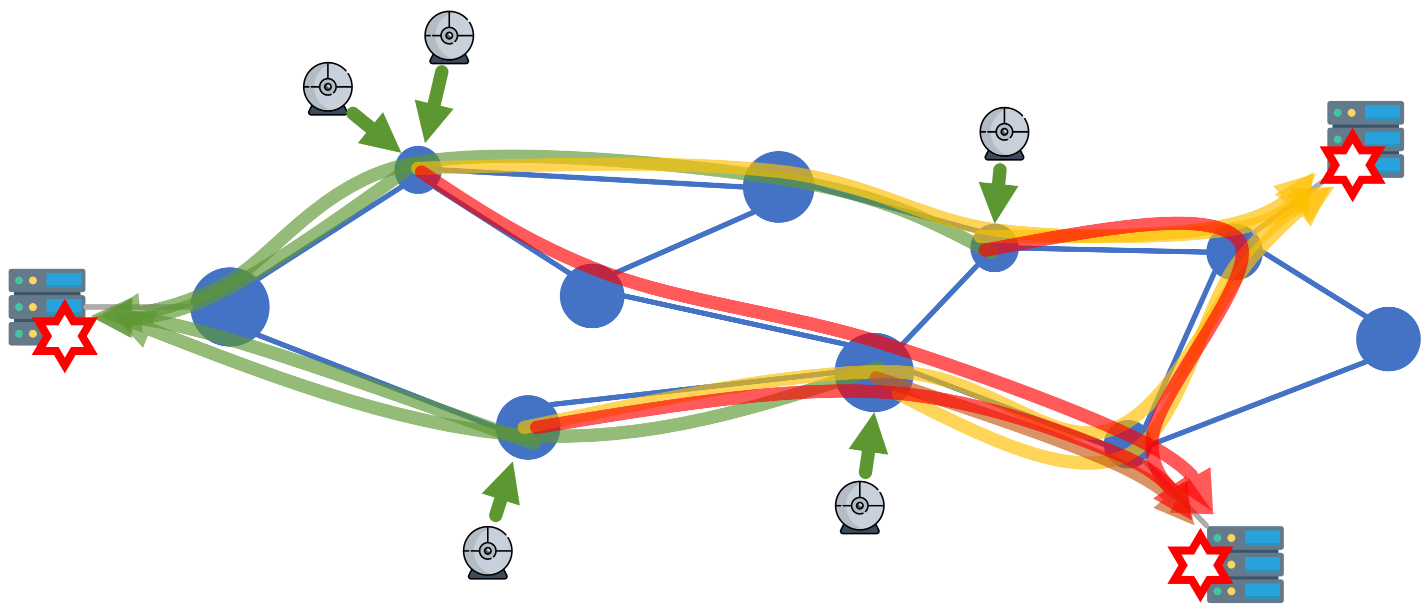
- SLA requests arrive randomly
- Each requests for a fixed period
  - Application owner (AO) sets a max price
- Each must be acc/rej at once

Request 1

Request 2

Request 3

Request 4



Outlines

Background and Motivation

System Modeling

Solution Design

Performance Evaluation

Discussions, Future Work and Conclusions

Offline Social Welfare Maximization

- **Goal:** accept as many SLA requests maximizing sum of valuation of all requests. Problem is NP-hard!

Offline social welfare optimization

$$\max_{\Phi, Z} \quad S(Z) = \sum\nolimits_j v_j \cdot \zeta_j \tag{7}$$

Social welfare

$$\text{s.t.} \quad \sum\nolimits_{h \in \mathcal{F}} x_j(s,h) \geq \zeta_j, \quad \forall j, s \in \mathcal{S}_j; \tag{7a}$$

App placement

$$\sum_j \sum_{p \in \mathcal{P}_l \cap \mathcal{P}^j} \tau(T,j) \cdot f_j(p) \leq b_l, \quad \forall l \in \mathcal{L}, T \in \mathcal{T}; \tag{7b}$$

Routing & bw allocation

$$\sum\nolimits_j \tau(T,j) \cdot \mathbf{B}_j \cdot r_k^j \cdot y_j(h) \leq c_k^h, \quad \forall h \in \mathcal{F}, k \in [\mathbf{K}], T \in \mathcal{T}; \tag{7c}$$

Resource allocation

$$(2) \textit{ for } \forall A_j \in \mathcal{A}_{\mathsf{sync}}; \qquad (3) \textit{ for } \forall A_j \in \mathcal{A}_{\mathsf{cent}}. \tag{7d}$$

Application type-specific constraints

$$y_j(h) = x_j(s,h), \quad \forall h \in \mathcal{F}, s \in \mathcal{S}_j. \tag{2}$$

Synchronous application

$$\textit{or} \quad y_j(h) \in \{0,1\}, \quad \forall h \in \mathcal{F}. \tag{3}$$

Centralized application

Online Competitive Social Welfare Maximization

- **Goal:** accept requests as each one comes, and decide payment.
  - Technique: primal-dual online competitive design [Buchbinder & Naor, 2009]

Definition 3. *A* θ-competitive mechanism *achieves at least* S^{opt}/θ *in social welfare, while satisfying all constraints.* □

Steps of Online Competitive Algorithm

1. Set prices σ_l and $\sigma_{n,k}$ exponential to utilization.
2. Find min-price provisioning scheme for request i.
3. If actual price > AO max price: Reject; else: Accept and charge actual price.

Competitive Ratio

Under **modest assumptions** on the resource/bandwidth and valuation per request, the online algorithm is $O(\log n)$-competitive, which can be shown tight following existing work (omitted due to page limit).

Issue: *How to calculate **min-price provisioning** while satisfying SLA? => NP-hard!*

N. Buchbinder and J. (Seffi) Naor, "The Design of Competitive Online Algorithms via a Primal—Dual Approach," Found. Trends® Theor. Comput. Sci., vol. 3, no. 2–3, pp. 93–263, mar 2009.

FPTAS for Min-Price Provisioning

- **Goal:** given a request and current link/node prices, calculate a min-price provisioning scheme satisfying end-to-end delay.
- **Observation 1:** the min-price provisioning scheme is always single-path based.
- **Observation 2:** we can omit bandwidth & resource capacities when provisioning one request.

Theorem 3: *Min-price provisioning can be solved by an extension of an existing Delay-Constrained Least-Cost (DCLC) routing algorithm, which is a fully polynomial-time approximation scheme (FPTAS).*

- *FPTAS: Given arbitrary ϵ, finds an $(1+\epsilon)$-approximation of the min-price provisioning scheme within time polynomial to $1/\epsilon$.*

Theorem 4: *Online algorithm + FPTAS =* $O((1+\epsilon)\log n)$ *competitive ratio.*

N. Buchbinder and J. (Seffi) Naor, "The Design of Competitive Online Algorithms via a Primal—Dual Approach," Found. Trends® Theor. Comput. Sci., vol. 3, no. 2–3, pp. 93–263, mar 2009.

Advantages of Our Solution

Computational advantages

- Competitive social welfare, in polynomial time
- Truthfulness: AO won't bid arbitrarily to manipulate prices
- Individual rationality & budget balance: no one loses money

Practical advantages

- AO's knowledge of the edge infrastructure not needed
- EP has full control over provisioning and tunable pricing
- Result applies to Centralized, Synchronous or Asynchronous apps

Implications

- First offline truthful competitive mechanism as well

N. Buchbinder and J. (Seffi) Naor, "The Design of Competitive Online Algorithms via a Primal—Dual Approach," Found. Trends® Theor. Comput. Sci., vol. 3, no. 2–3, pp. 93–263, mar 2009.

Outlines

Background and Motivation

System Modeling

Solution Design

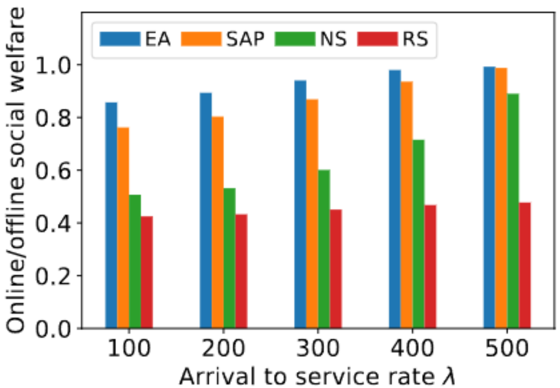
Performance Evaluation

Discussions, Future Work and Conclusions

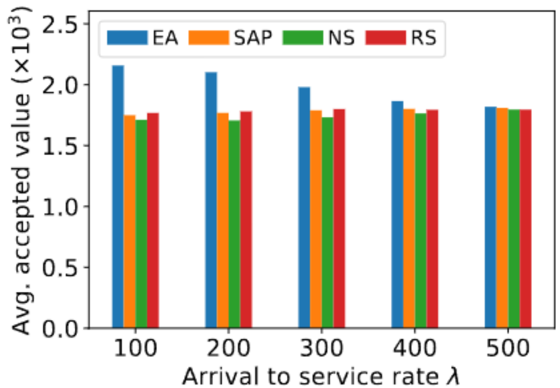
Simulation Settings – Demand & Network

- Settings
  - Simulated edge network
    - 20 mesh-connected APs with 5 edge nodes (3 types of resources per node)
    - Waxman topology with $\alpha = \beta = 0.6$
    - 1.2Gbps and 10-50ms links, 3-10Gbps computation capacity (normalized)
  - Synthetic application requests
    - 1000 Poisson arrival requests with arrival rate of 300
    - 5-10 sources per request, 3-10Mbps traffic per source, 25-75ms delay bound
    - AO valuations set based on assumptions
  - $\epsilon = 0.5$ (FPTAS accuracy)
  - Comparisons
    - SAP (FPTAS from prior work), ODA (offline delay-agnostic upper bound)
    - Random Selection (RS) and Nearest Selection (NS) heuristics

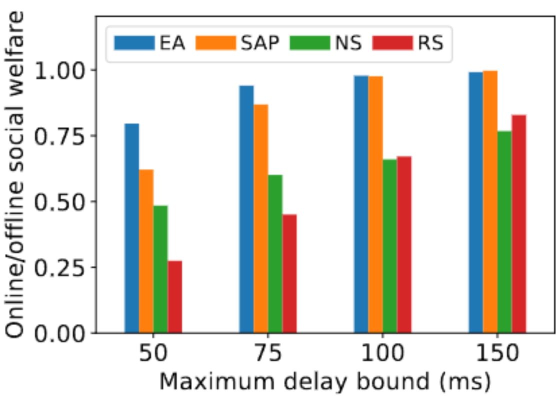
Comparison to Baselines



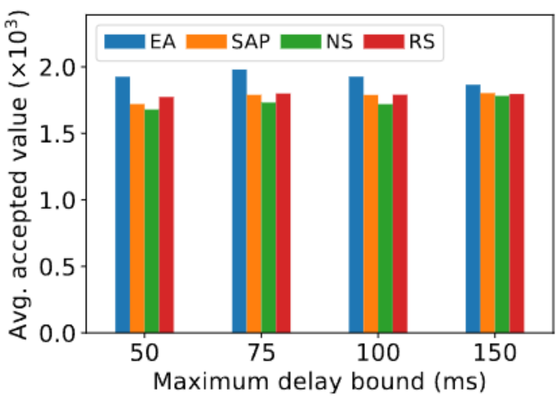
(a) Social welfare with varying load



(b) Average value with varying load



(c) Social welfare with varying maximum delay bound



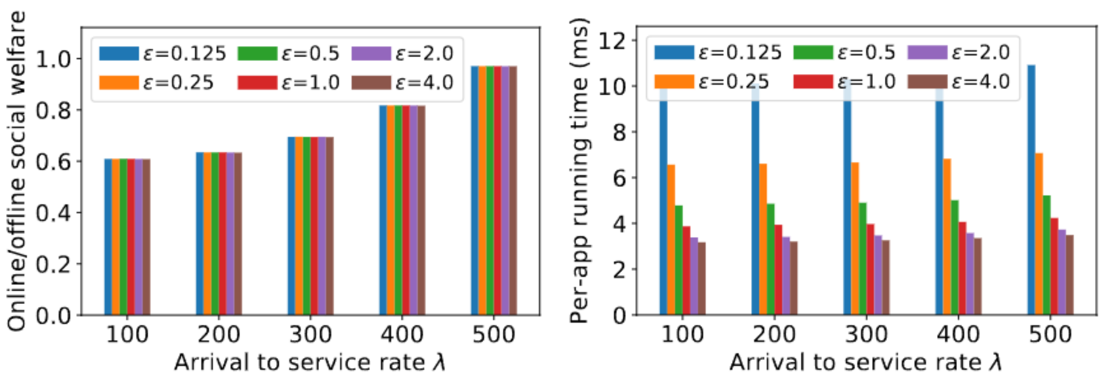
(d) Average value with varying maximum delay bound

Note: *Social welfare normalized to offline upper bound ODA.*

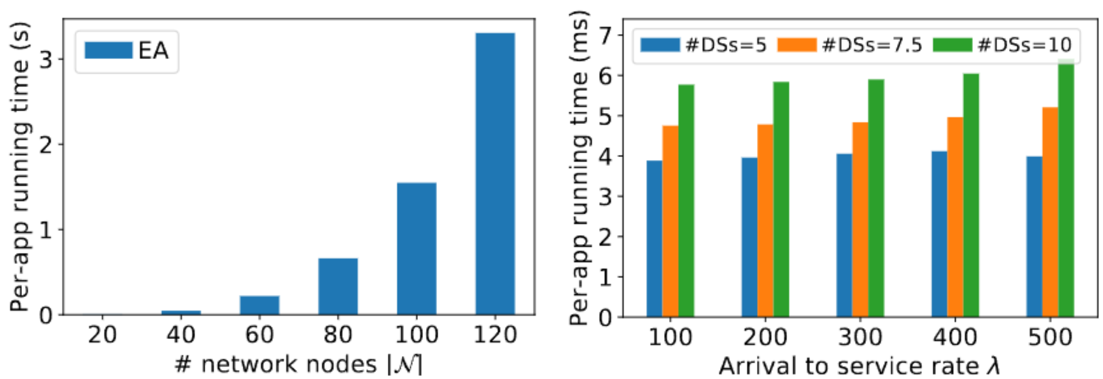
EA (our mechanism) can:

1. Achieve **superior social welfare**
2. Accept requests with **higher average values**

Scalability



(a) Social welfare vs. ϵ (b) Running time vs. ϵ



(c) Running time vs. # APs (d) Running time vs. # data sources

λ	100	200	300	400	500
EA	5.62	5.66	5.68	5.62	5.55
SAP	10110.29	11658.24	14152.57	17681.29	21985.57

TABLE III: Running time per request (ms) for EA and SAP.

EA can achieve scalability via setting a loose ϵ, without sacrificing performance.

EA achieves superior scalability compared to SAP heuristic.

More results in our paper

Outlines

Background and Motivation

System Modeling

Solution Design

Performance Evaluation

Discussions, Future Work and Conclusions

Other Perspectives, Conclusions

- So far, we've talked about
  - Application SLA provisioning + dynamic pricing

A unique combination of online mechanism and optimization algorithm

- What could be improved
  - More realistic applications: microservice, FL, …
  - Wireless characteristics
  - Demand estimation and prediction
  - Reliability and robust provisioning
  - SLA monitoring and verification
  - Improved optimization methods
  - Improved statistical & learning-based methods

Modeling Perspective

SLA Perspective

Algorithmic Perspective

- **Conclusions:** building an app-centric edge ecosystem.

This research was supported in part by NSF grants 2007391, 2007469, and 2045539. The information reported here does not reflect the position or the policy of the funding agency.

Thank you very much!

Q&A?