A New Class of Polynomial Activation Functions of Deep Learning for Precipitation Forecasting

Jiachuan Wang*, Lei Chen*, Charles Wang Wai Ng*

*The Hong Kong University of Science and Technology, Hong Kong, China

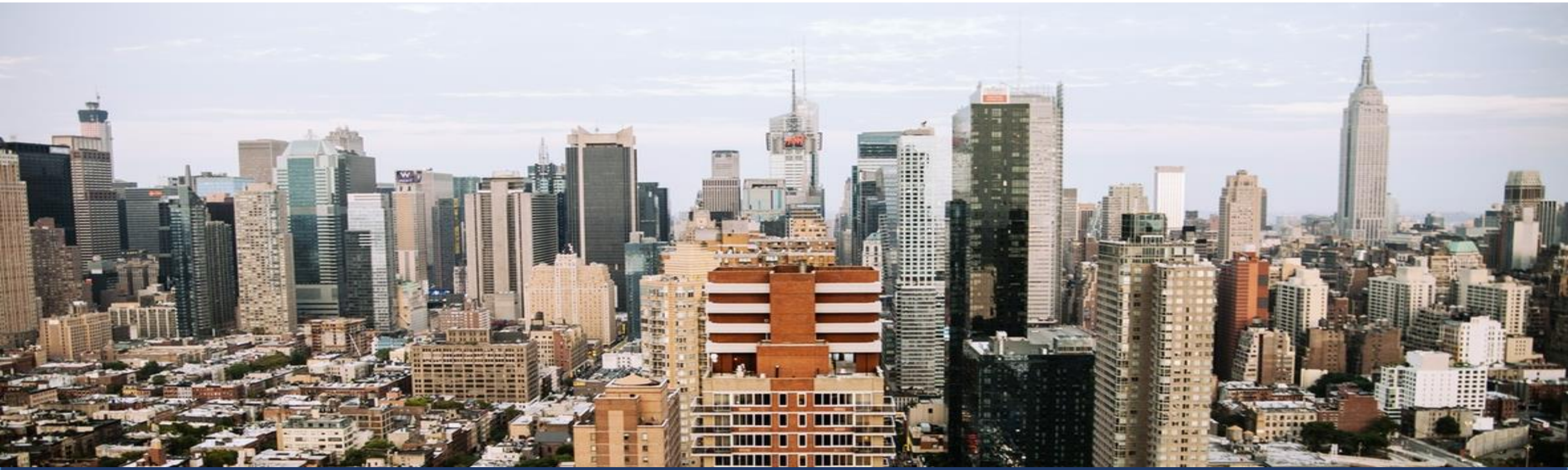
Outline

- Motivation
- Problem Formulation
- Methods
- Evaluations

Background

Importance of Rainfall Forecasting

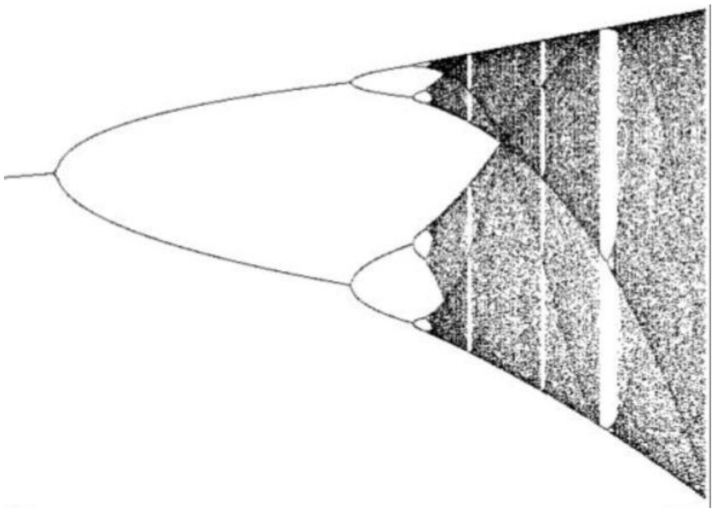
- Prevention of natural disasters
- Management of infrastructure systems
- Maintenance of safe environment



Hardness – Why it is difficult?

Chaotic system

Tiny initial error → **Huge** output error



Convection → heavy rain

But *"convection is not explicitly resolved"*

*[1]~"Sub-daily precipitation **extremes** are often produced by **convective events**, but conventional global and regional climate models are not able to simulate such events well because of limited spatial and temporal resolution and because **convection is not explicitly resolved**"*

[1] Zhang X , Zwiers F W , Li G , et al. Complexity in estimating past and future extreme short-duration rainfall[J]. Nature Geoence, 2017, 10(4):255-259.

Route Planning for Shared Mobility

- **Large** amount of **dynamically** arriving requests

- **Large** amount of workers


Route Planning for Shared Mobility

- **Large** amount of possible route allowing share

- **Limited** response time

Route Planning for Shared Mobility

- **Large** amount of **dynamically** arriving requests
- **Large** a
- **Large** a … e
- **Limited** response time

effective / efficient
route planning strategy

Keep the Balance of Demand-Supply

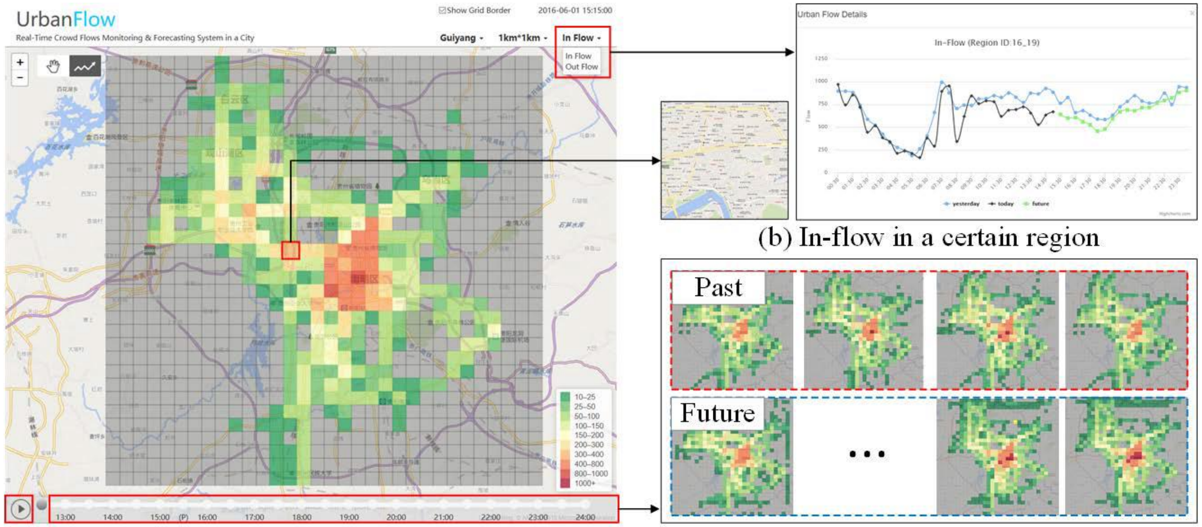
Great profit loss from ***unmatched*** distribution of ***demand*** and ***supply***

- **Rush hour** **(ridesharing)**
  - Morning: rural areas → center of city
  - Evening: center of city → rural areas

- **Lunch and supper time** **(food delivery)**
  - Tons of orders sent to business central

Prediction of Demand

Derive demands in different areas/timesteps accurately using **spatial temporal** prediction. (e.g. DeepST*)



(a) User interface of UrbanFlow

(b) In-flow in a certain region

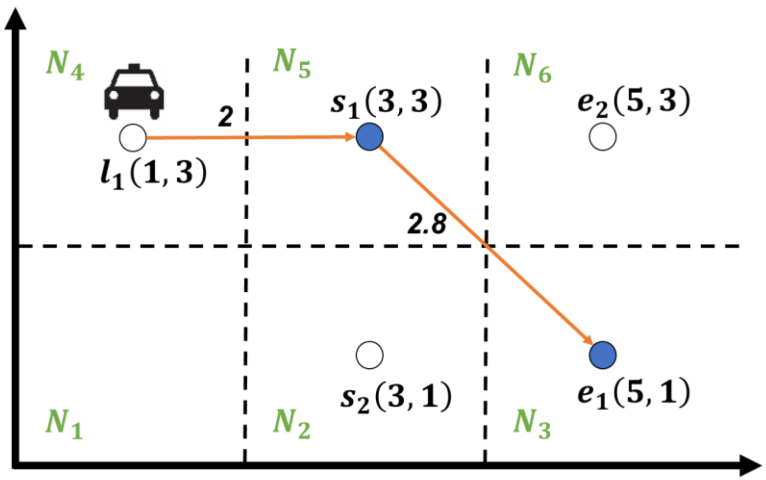
(c) Heatmaps in past & future

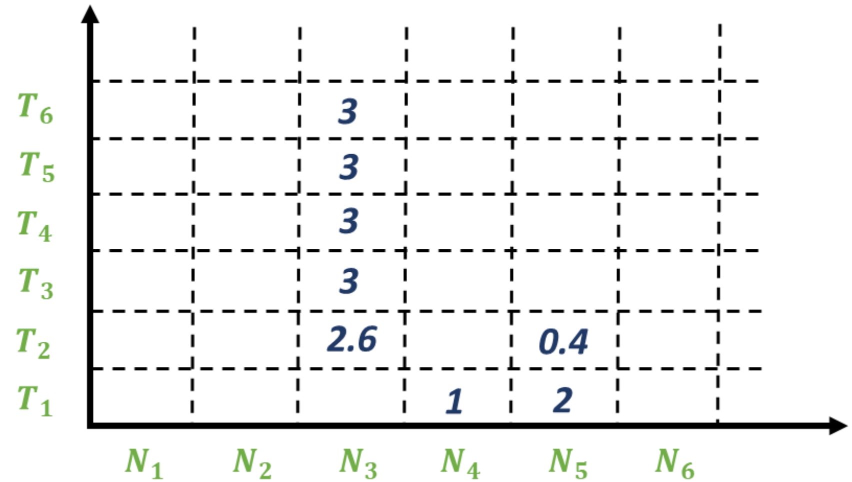
*Figure is Provided by Junbo Zhang

How to use it to benefit route planning for shared mobility?

Supply Organization

In each **area** and **time span**, larger total **time duration** of workers leads to higher probability to serve a request.



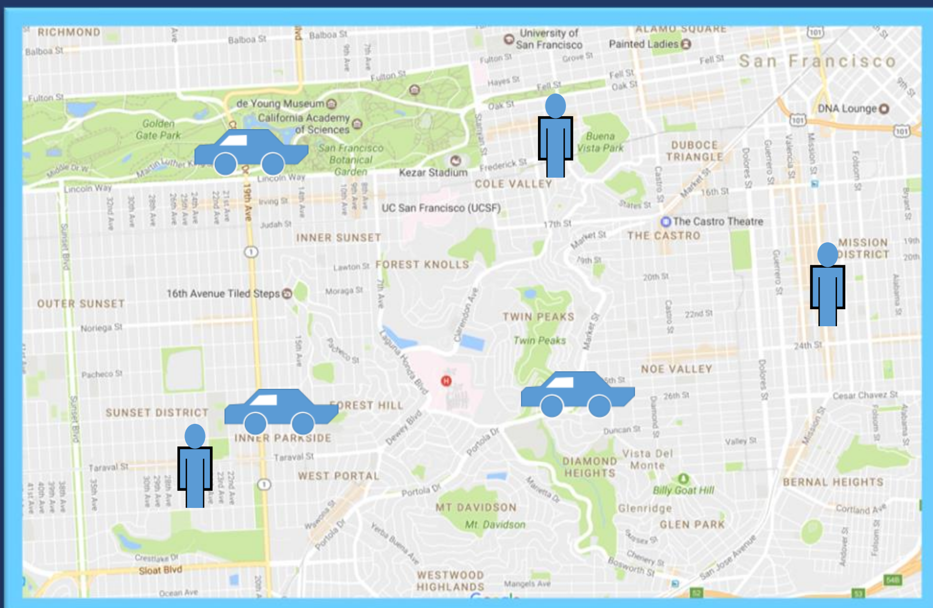


Different **route planning** strategy

→ → Different **distribution of supply**

Organize **supply** according to **demand** to maximize profit.

Motivation



Design algorithm to **improve the effectiveness** of *route planning* for shared mobility through:

Evaluating the effect of **supply** during route planning based on **demand**.

The **overall profit** of the platform is **improved**.

Outline

- Motivation
- Problem Formulation
- Algorithms
- Evaluations

Workers and Requests

Workers

$$W = \{w_1, w_2, \dots, w_n\}$$

w_i

- current location l_i
- capacity a_i

Requests

$$R = \{r_1, r_2, \dots, r_m\}$$

r_j

- start/end loc. s_j/e_j
- release time tr_j
- deadline td_j
- rejection penalty p_j
- capacity a_j

Workers and Requests

Workers

$W = \{w_1, w_2, \dots, w_n\}$

w_i

- current location l_i
- capacity a_i

Requests

$R = \{r_1, r_2, \dots, r_m\}$

r_j

- start/end loc. s_j/e_j
- release time tr_j
- deadline td_j
- rejection penalty p_j
- capacity a_j

Workers and Requests

Workers

$W = \{w_1, w_2, \dots, w_n\}$

Requests

$R = \{r_1, r_2, \dots, r_m\}$

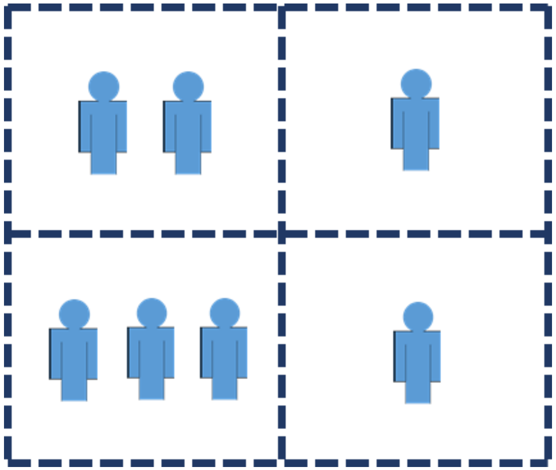
Route S_i, a sequence of s_j/e_j

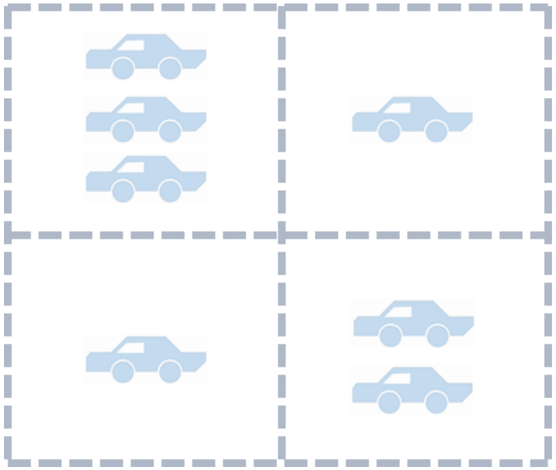
w_i → Planning routes to serve requests. → r_j

Existing work* handles the **distance** related cost.

Quantize Demand-Supply Balance

1. Demand number map (DN)
   - **Number of requests** in each time span and area
   - Predicted using deep learning model*





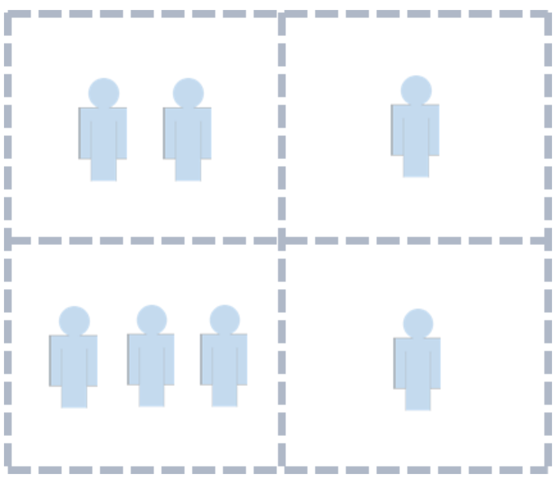
Quantize Demand-Supply Balance

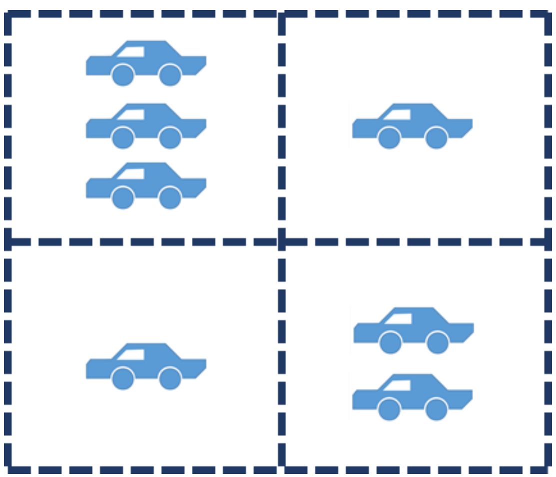
1. Demand number map (DN)
2. Supply number map (SN)
   - **Number of worker** in each time span and area
   - Updated according to route planning

Route plan affects Supply Contribution

Supply number

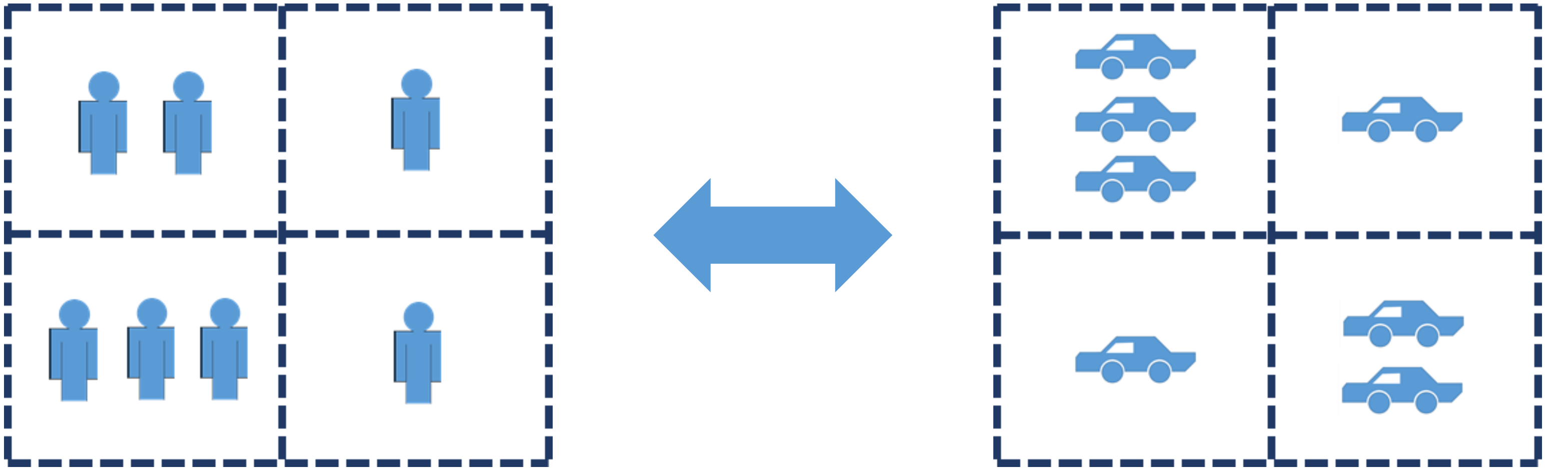
$$SN(N_x, T_y) = \sum_{w_i \in W} \sum_{k=1}^{|S_{w_i}|-1} \boxed{SC_{xy}(l_k, l_{k+1}, arr_i[l_k])}$$





Quantize Demand-Supply Balance

1. Demand number map (DN)
2. Supply number map (SN)
3. Demand-Supply Balance Score (DSB)
   - Each route plan affects future supply and balance
   - Statistically analyze expected **profit** of the balance



Quantize Demand-Supply Balance

1. Demand number map (DN)
2. Supply number map (SN)
3. Demand-Supply Balance Score (DSB)
    - Each route plan affects future supply and balance
    - Statistically analyze expected **profit** of the balance

Local Balance $LB(\lambda, sn)$
(Expected # of matchings)

Local demand (λ) and supply (sn)

$$= E(Y) = \sum_{k=0}^{\lfloor sn \rfloor - 1} k \frac{\lambda^k}{k!} e^{-\lambda} + \sum_{k=\lfloor sn \rfloor}^{\infty} \lfloor sn \rfloor \frac{\lambda^k}{k!} e^{-\lambda} \quad (3)$$

$$= \sum_{k=0}^{\lfloor sn \rfloor - 1} (k - \lfloor sn \rfloor) \frac{\lambda^k}{k!} e^{-\lambda} + \sum_{k=0}^{\infty} \lfloor sn \rfloor \frac{\lambda^k}{k!} e^{-\lambda}$$

$$= \sum_{k=0}^{\lfloor sn \rfloor - 1} (k - \lfloor sn \rfloor) \frac{\lambda^k}{k!} e^{-\lambda} + \lfloor sn \rfloor$$

Analyzed based on ***Poison distribution***

$$DSB = \sum_{x=1}^{|\mathcal{N}|} \sum_{y=1}^{|\mathcal{T}|} \beta_y \left({sn'}_{xy} - sn_{xy} \right) \cdot \boxed{\Delta_{LB}(\lambda_{xy}, sn_{xy})}$$

Change of LB

Demand-Aware Route Planning (DARP) Problem

Given a set of workers W, a set of requests R, a demand number map DN, the DARP Problem is to find the sets of routes S for all the workers to minimize **Demand-Aware Cost (DAC)**:

Cost from workers' moving distances

Cost from Demand-Supply Balance

Cost from rejection

$$DAC(W, R, DN) = \boxed{\alpha \sum_{w_i \in W} D(S_{w_i})} - \boxed{DSB(\beta, S, DN)} + \boxed{\sum_{r_j \in \bar{R}} p_j}$$

Such that:

1. at any time the total capacity of requests of any worker should not exceed its **capacity** a_i;
2. each request meets its **deadline**;
3. An assigned request cannot be assigned to another; a rejected request cannot be revoked.

Demand-Aware Route Planning (DARP) Problem

Given a set of workers W, a set of requests R, a demand number map DN, the DARP Problem is to find the sets of routes S for all the workers to minimize **Demand-Aware Cost (DAC)**:

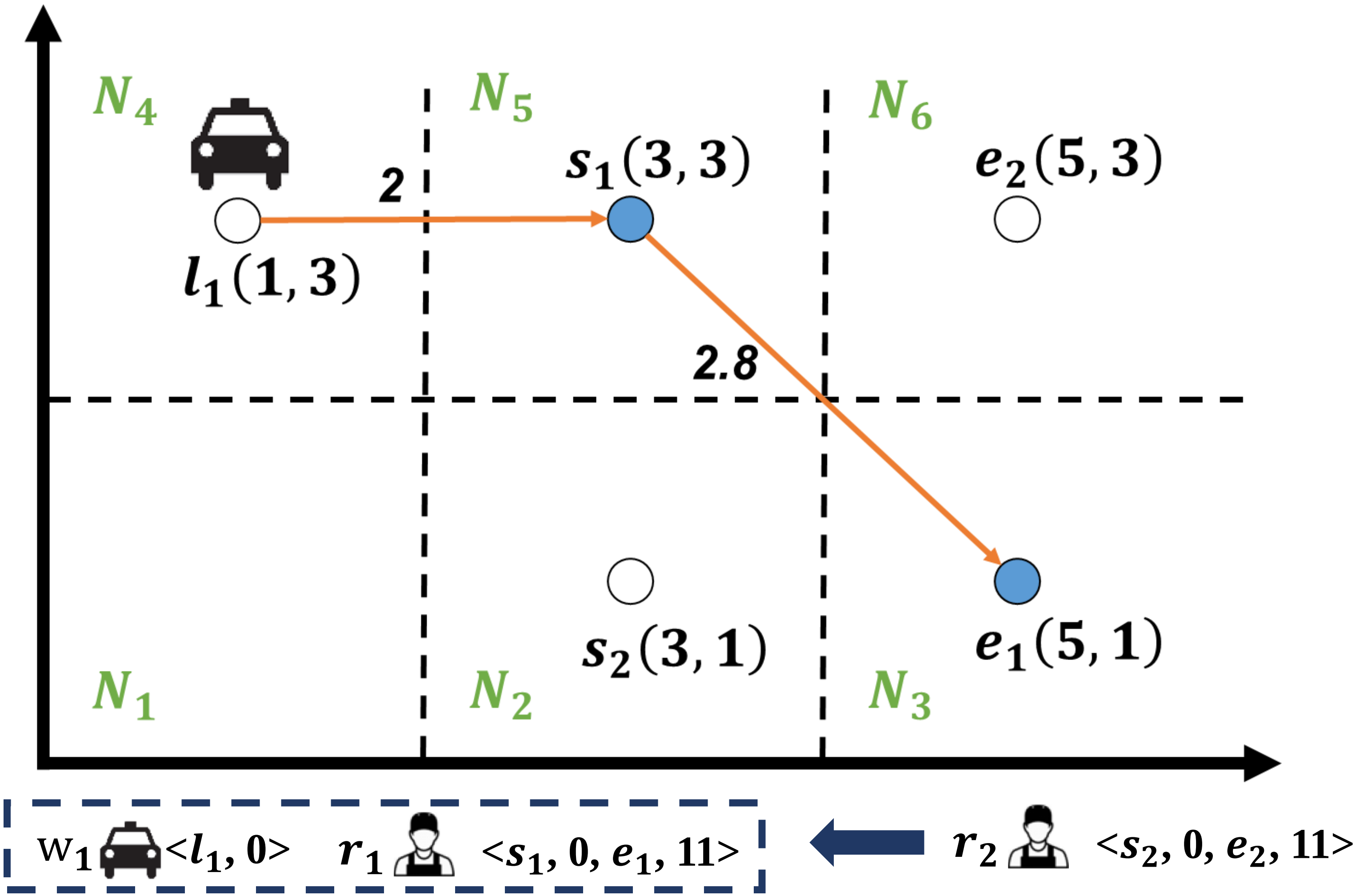
$$DAC(W, R, DN) = \underbrace{\alpha \sum_{w_i \in W} D(S_{w_i})}_{\text{Cost from workers' moving distances}} - \underbrace{DSB(\beta, S, DN)}_{\text{Cost from Demand-Supply Balance}} + \underbrace{\sum_{r_j \in \bar{R}} p_j}_{\text{Cost from rejection}}$$

We prove the DARP problem is **NP-hard** by reducing it from basic route planning problem* for shareable mobility services. We further show that **no** randomized or deterministic algorithm can guarantee a **constant Competitive Ratio**

*Yongxin Tong, et al.: A Unified Approach to Route Planning for Shared Mobility. Proc. VLDB Endow. 11(11): 1633-1646 (2018)

Running Example

Time spans $[0\sim 3, 3\sim 6, \cdots, 15\sim 18]$
Areas $[N_1, N_2, \cdots, N_6]$



How to derive cost if we assign request r_2?

Running Example

Time spans $[0\sim3, 3\sim6, \cdots, 15\sim18]$
Areas $[N_1, N_2, \cdots, N_6]$

Table 1: Supply Number Map

$\mathcal{T}$ \ $\mathcal{N}$	N_1	N_2	N_3	N_4	N_5	N_6
T_1	1.7	3.8	2.5	2.3	0.5	1.3
T_2	3.3	2.1	1.7	1.1	3.2	2.9
T_3	3.5	3.3	2.0	0.7	3.8	1.4
T_4	3.6	1.3	2.4	3.0	1.2	2.6
T_5	0.5	2.5	1.4	1.3	1.6	2.3
T_6	3.4	2.0	1.0	3.7	2.2	3.8

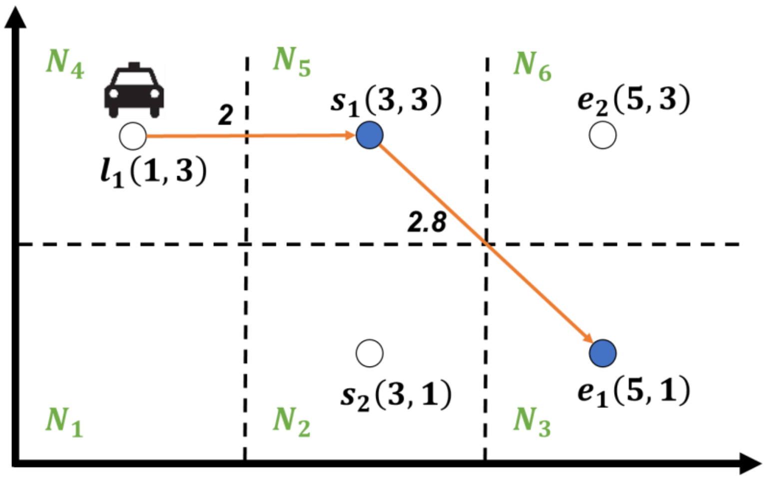
Table 2: Demand Number Map

$\mathcal{T}$ \ $\mathcal{N}$	N_1	N_2	N_3	N_4	N_5	N_6
T_1	2	3	5	2	4	3
T_2	3	2	4	2	3	2
T_3	3	3	4	2	2	2
T_4	3	1	4	3	2	2
T_5	4	2	4	3	1	3
T_6	3	2	3	4	2	2

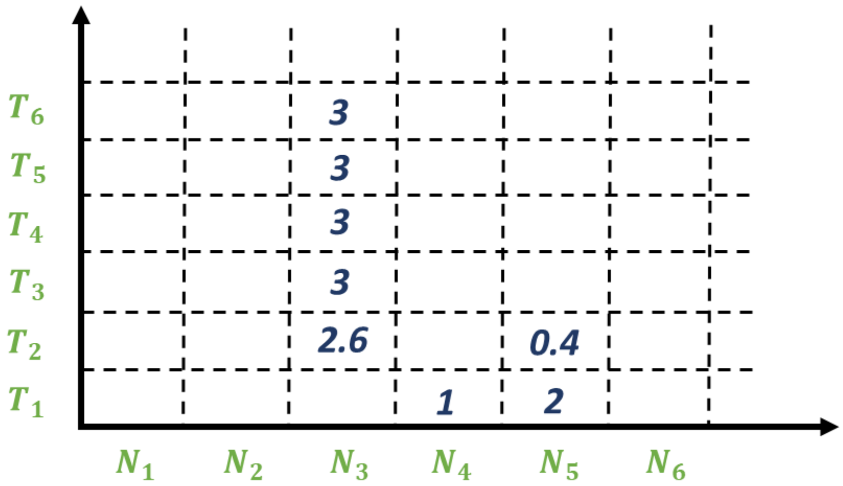
Supply number map **(SN)** and Demand number map **(DN)** are required for cost of Demand-Supply Balance **(DSB)**

Running Example

Route before insertion



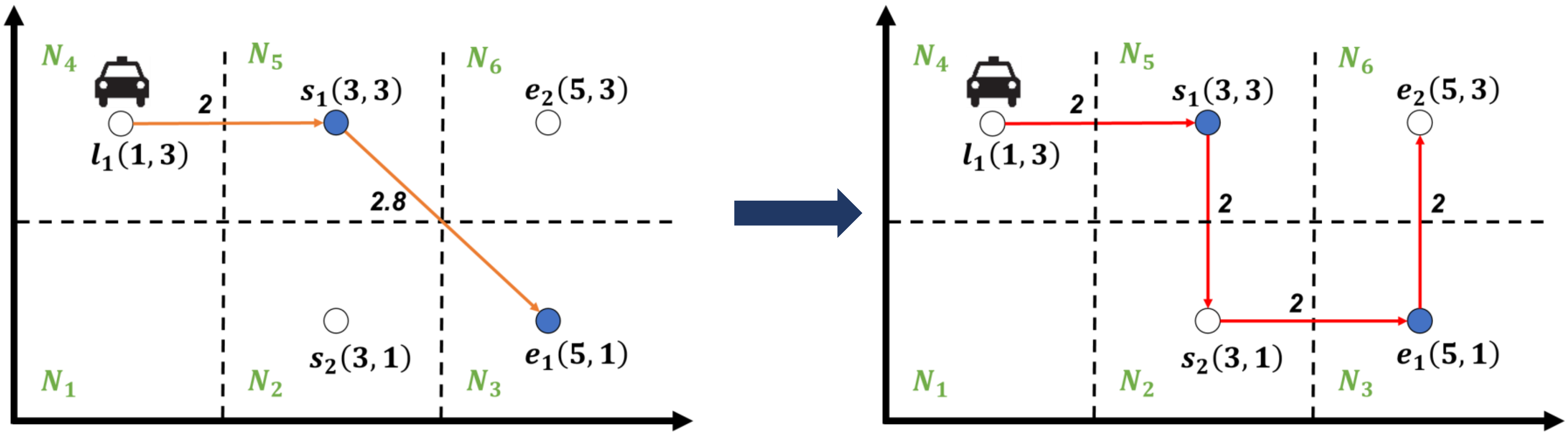
Time duration of $\boldsymbol{w_i}$ in spatiotemporal cells of **SN** before insertion



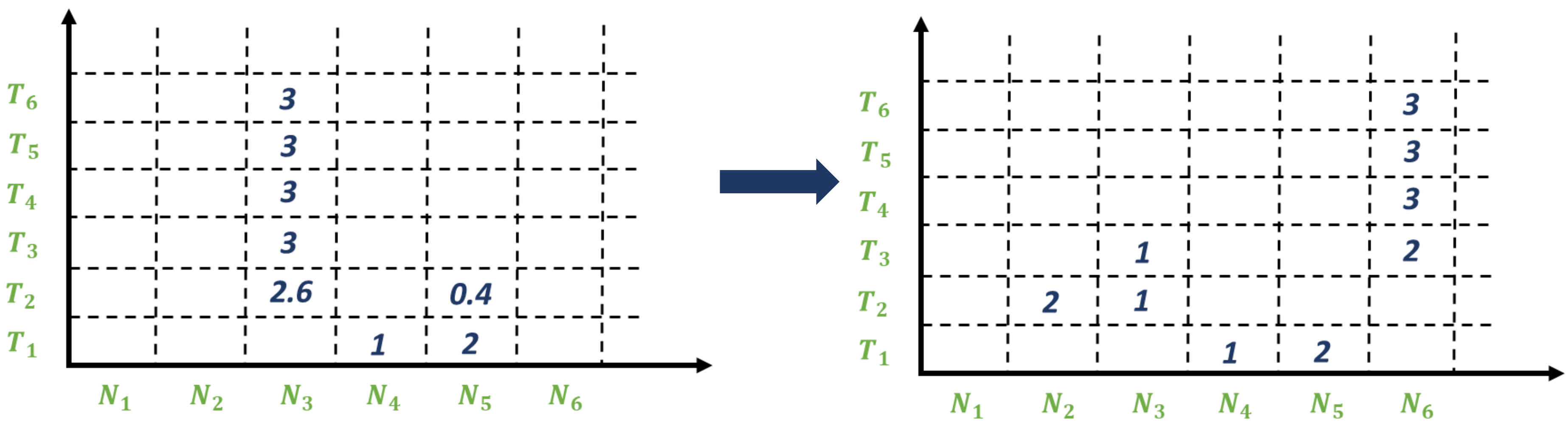
Running Example

A possible insertion

Route before/after insertion



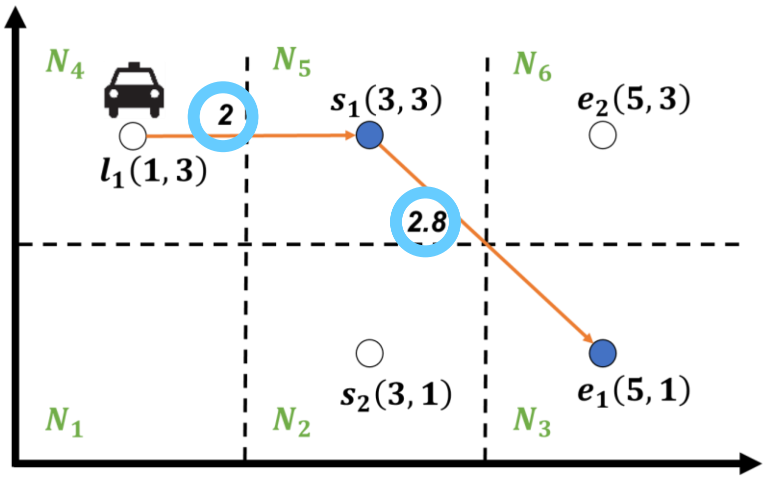
Time duration of w_i in spatiotemporal cells of **SN** before/after insertion

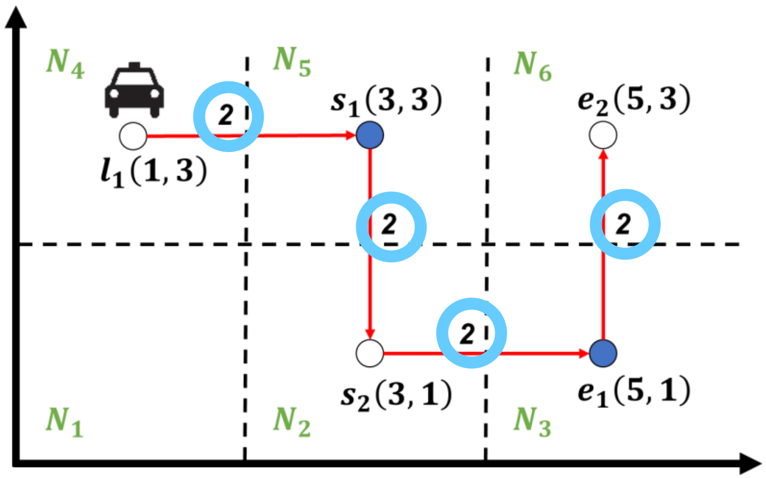


Running Example

A possible insertion

Route before/after insertion





Shortest distance to finish original route
2+2.8

Shortest distance to finish inserted route
2+2+2+2

Cost from workers' moving distances
is derived according to the difference of finishing time

Running Example

A possible insertion

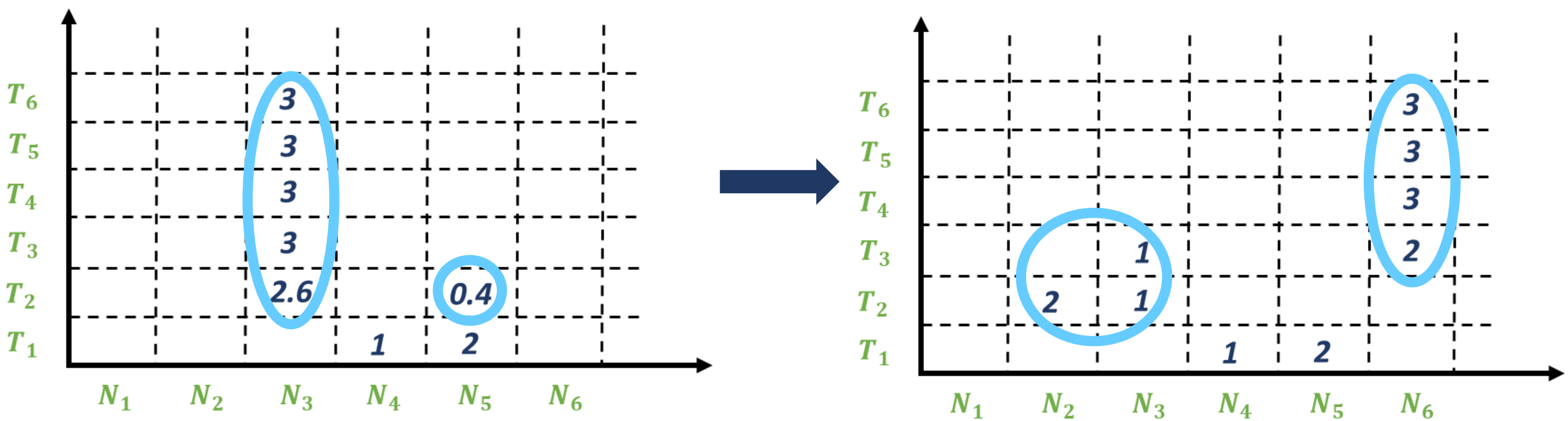
Cost from Demand-Supply Balance (DSB)

Predicted Demand Number Map (DN)

Supply Number Map (SN)

Change of time duration affects the supply in each spatiotemporal cell

Time duration of w_i in spatiotemporal cells of **SN** before/after insertion



Running Example

Cost from Demand-Supply Balance (DSB)

Total cost of a valid insertion

Cost from workers’ moving distances is derived according to the difference of finishing time

Outline

- Motivation
- Problem Formulation
- Algorithms
- Evaluations

Proposed Approaches

To solve the DARP problem, we proposed

Insertion algorithm (single request)

- **Basic** insertion
- **Dynamic programming**-based insertion

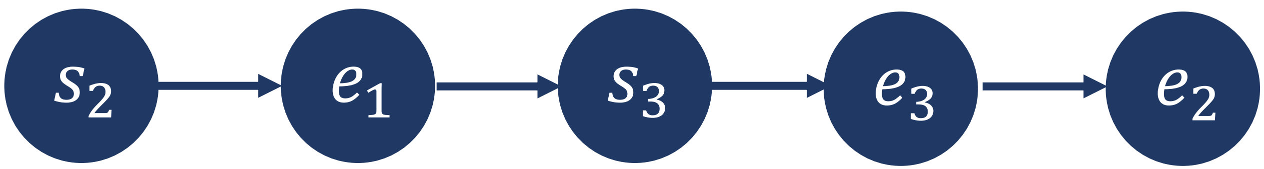
Solution for DARP problem

- Insertion-based **dual-phase framework**

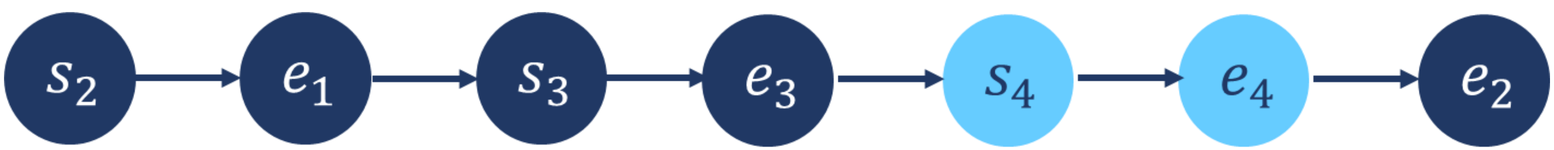
Insertion

Insertion: one request → one worker's route

effective and efficient approach



Original nodes are in the **same order** (search space ↓):



Insertion

Naturally: $\boldsymbol{O(N^3)}$ time complexity

Distance-related cost: $\boldsymbol{O(N)}$.

Existing work* reduces its cost as: additional distance from inserting source and destination are **separable**.

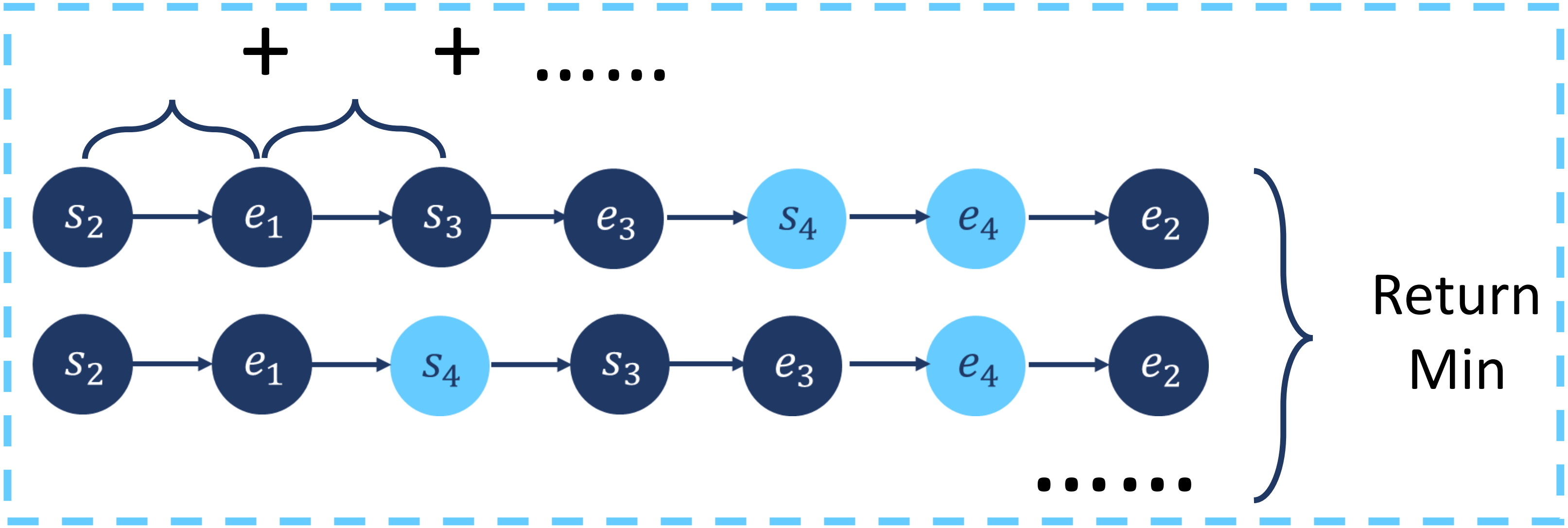
Demand supply balance cost: previous $\boldsymbol{O(N^2)}$ and $\boldsymbol{O(N)}$ algorithms are **not** appliable

Detour of inserting source affects all the supply

from latter nodes.

The Basic Insertion Algorithm ($O(N^3)$)

1. **Enumerate** insertion pairs for a length-N route
   - $O(N^2)$ cases
2. For **each** new route, derive the cost
   - $N + 1$ small paths. **Calculate** and **sum up** them cost $\boldsymbol{O(N)}$



3. Return the plan with **lowest** cost (Greedy)

The DP-Based Insertion Algorithm ($O(N^2)$)

1. **Enumerate** insertion pairs

- $O(N^2)$ cases

2. For each new route, derive the cost in $O(1)$ time

- **Distance**-related cost in $O(1)$: studied*
- Cost from Demand Supply Balance (**DSB**): how?

The DP-Based Insertion Algorithm ($O(N^2)$)

1. Dynamically derive a ***check-up table*** in $\boldsymbol{O(N)}$ ***time***
   - Derive the maximum time to delay for each node.
   - Divide it into a discretized space. Increasing **DSB** is stored with **time delay**.

2. Enumerate insertion pairs
   - $O(N^2)$ cases

3. For each new route, derive the cost ***in*** $\boldsymbol{O(1)}$ ***time***
   - Distance-related cost in O(1): studied*
   - Cost from DSB: ***check in*** $\boldsymbol{O(1)}$ ***time according to time delay***

4. Return the plan with lowest cost

The DAIF framework

- Assign requests **one-by-one**
- Quickly derive a ***lower bound*** of cost for each worker
  - In $\boldsymbol{O}(\boldsymbol{N})$ time + only ***1*** shortest path query.
  - Existing work* derive the lower bound for ***distance cost***
  - We efficiently derive a lower bound for ***balance cost*** based on the property of Demand-Supply-Balance cost
- ***Sort*** → Calculate ***exact*** cost → ***Prune & insert***
  - Derive **exact** cost for each worker ordered by lower bound
  - If the lower bound is larger than current minimal cost, ***safely prune*** all the rest workers

**Yongxin Tong, et al.: A Unified Approach to Route Planning for Shared Mobility. Proc. VLDB Endow. 11(11): 1633-1646 (2018)*

Outline

- Motivation
- Problem Formulation
- Algorithms
- Evaluations

Experimental Setting

- Road Network
  - NYC (|V|=61,298, |E|=141,372)
- Real Datasets
  - Taxi Trips (2013) in NYC (427,093 trip records)
- Synthetic Dataset
  - Generated according to the distribution of NYC (452,116 trip records)

Experimental Setting

- Compared parameters
  - e_r: the deadline coefficient.
  - a_i: the capacity of workers.
  - α, β: the weight for distance/ balance cost.
  - γ: the factor that staying time duration of worker transfer to supply.
  - p_o: the ratio of penalty cost
  - $|W|$: number of workers
  - g: grid size

Parameters	Settings
Deadline Coefficient e_r	0.1, 0.2, **0.3**, 0.4, 0.5
Capacity a_i	2, **3**, 4, 7, 10
Distance Weight α	**1**
Balance Weight β	$\left[\boldsymbol{p_r^*, \frac{p_r^*}{e}, \frac{p_r^*}{e^2}, \cdots, \frac{p_r^*}{e^5}}\right]$
Supply Coefficient γ	**0.0016**
Penalty ratio p_o	**30**
Number of workers $\lvert W \rvert$	500, 1k, **3k**, 5k, 10k
Grid size g	1k×1k, **2k×2k**, 4k×4k

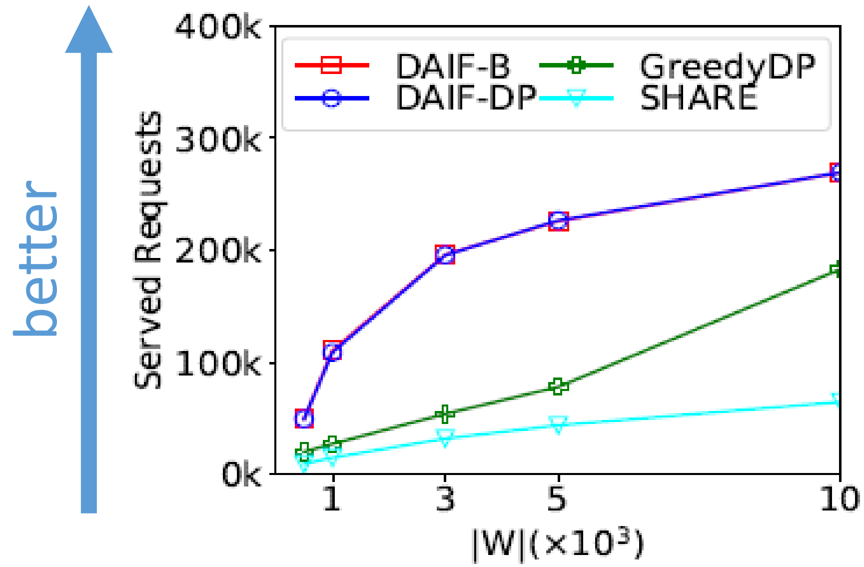
Experimental Setting

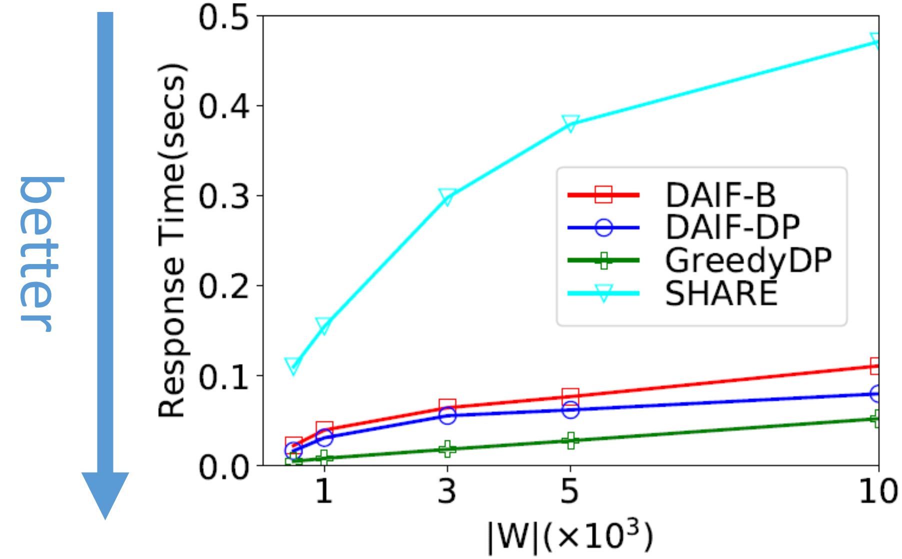
- Tested Algorithms
  - **GreedyDP***: the state-of-art route planning algorithm using insertion. No demand-related information is used.
  - **SHARE#**: It uses historical information of nodes to choose a route with a higher possibility to pick passengers along the route
  - **DAIF-B**: our DAIF framework using Basic insertion
  - **DAIF-DP**: our DAIF framework using DP-based insertion

*Yongxin Tong, et al.: A Unified Approach to Route Planning for Shared Mobility. Proc. VLDB Endow. 11(11): 1633-1646 (2018)

#Chak Fai Yuen, et al.: Beyond Shortest Paths: Route Recommendations for Ride-sharing. WWW 2019: 2258-2269

Experimental Results



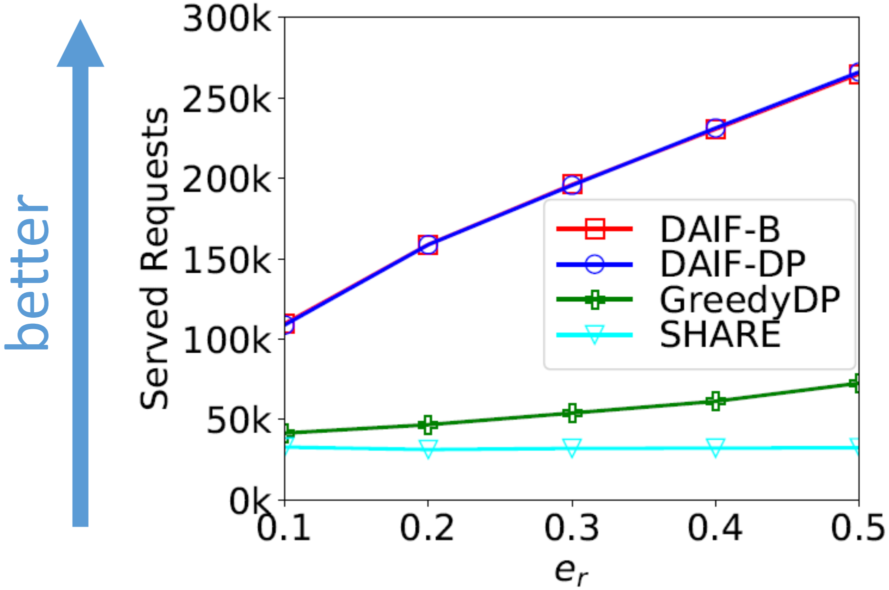


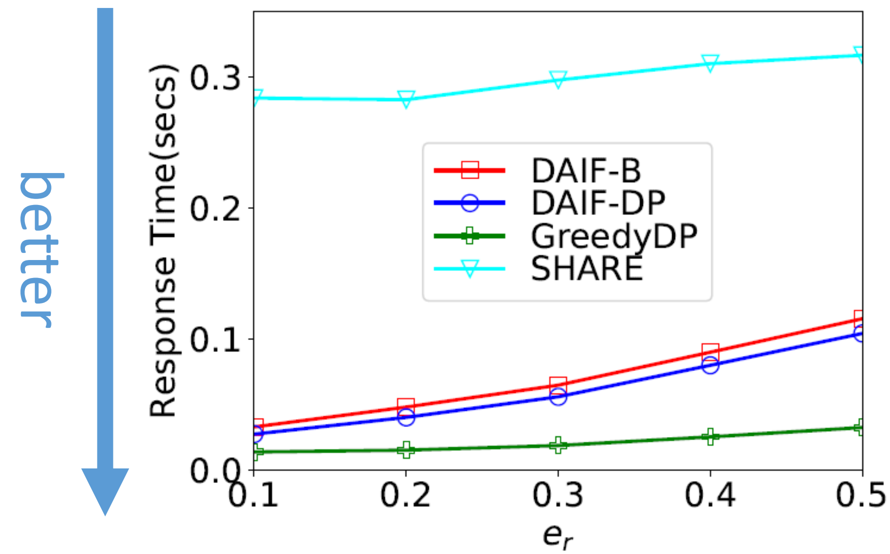
Serve ***47%*** to ***624%*** more requests

Response time < ***0.15s***

Performance of varying number of workers $|W|$

Experimental Results





Serve ***161.7%*** to ***718.1%*** more requests

Response time < ***0.15s***

Performance of varying deadline coefficient e_r

Thank You

Q&A

The code and datasets
https://github.com/dominatorX/DAIF