Mobility-Aware Dynamic Taxi Ridesharing

Zhidan Liu, Zengyang Gong, Jiangzhou Li, Kaishun Wu

Shenzhen University, Shenzhen, China

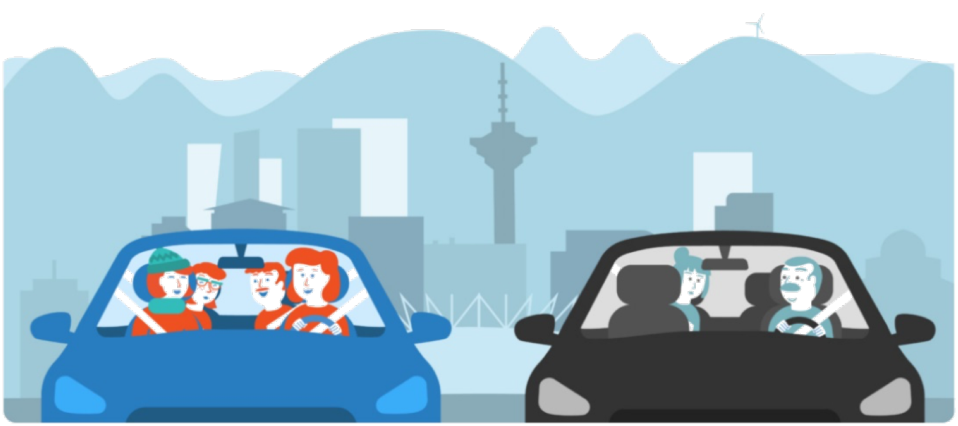
23. April, 2020

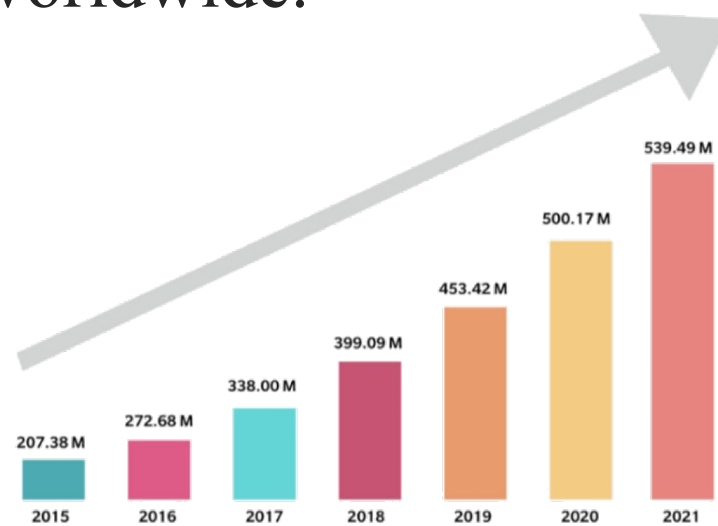
IEEE ICDE 2020

April 20th -24th, 2020 - Dallas, Texas, USA

Ridesharing

- *Ridesharing* allows multiple passengers with the similar itineraries and time schedules to share a vehicle.
  - It benefits many parts, e.g., alleviating traffic congestion, reducing energy consumption, etc.
- *Ridesharing* becomes popular worldwide.

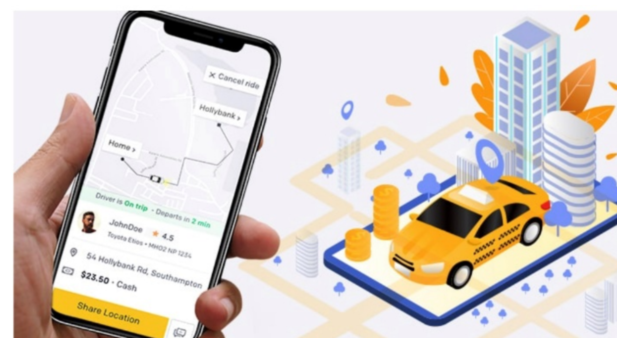




Taxi Ridesharing

- Taxi is an important transportation mode in all cities.
  - Taxis are widely available in a city, and are operating in 7x24.
  - Taxis can be either booked online, or hailed along the street.
- Taxi ridesharing becomes promising [1-3].

[1] S. Ma, Y. Zheng, O. Wolfson, et al. Real-time city-scale taxi ridesharing. IEEE Transactions on Knowledge and Data Engineering, 27(7):1782–1795, 2015.
[2] Q. Ma, Z. Cao, K. Liu, and X. Miao. QA-Share: toward an efficient QoS-aware dispatching approach for urban taxi-sharing. ACM Transactions on Sensor Networks, 16(2):1–21, 2020.
[3] W. Zhang, A. Shemshadi, Q. Z. Sheng, Y. L. Qin, X. Xu, and J. Yang. A user-oriented taxi ridesharing system with large-scale urban GPS sensor data. IEEE Transactions on Big Data, 1(1):1–14, 2019.



Modeling of taxi ridesharing systems

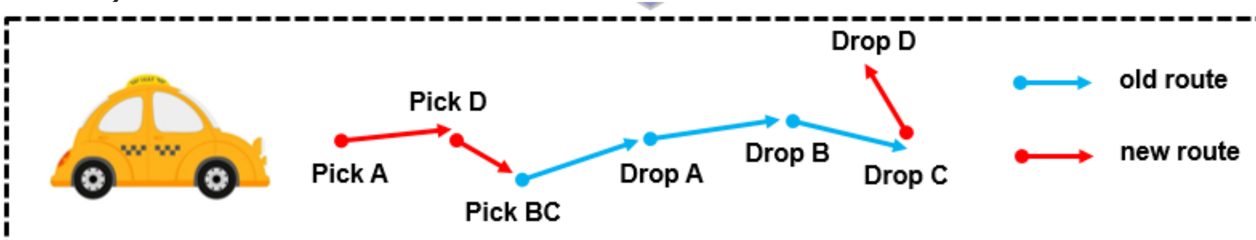
- Ride Request $r_i = < t_{r_i}, o_{r_i}, d_{r_i}, e_{r_i} >$
  - o_{r_i}, d_{r_i}: origin and destination
  - t_{r_i}, e_{r_i}: release time and delivery deadline

- Taxi Status $t_j = < loc_{t_j}, S_{t_j}, R_{t_j} >$

Taxi ridesharing is dynamic and challenging, since requests are generated on the fly and taxi schedules are continuously updated.

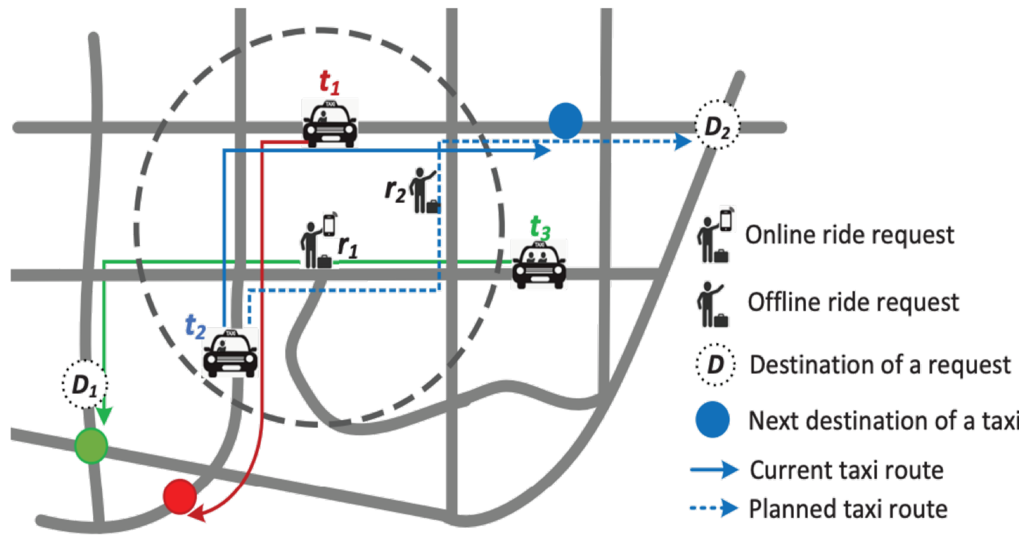
ns

  - R_{t_j} : taxi travel route



Existing solutions

- Existing systems [1-5] process a ride request in two stages
  - Candidate taxi searching
    - E.g., t_1 and t_2 for r_1
  - Ridesharing routing
    - Updating schedule/route



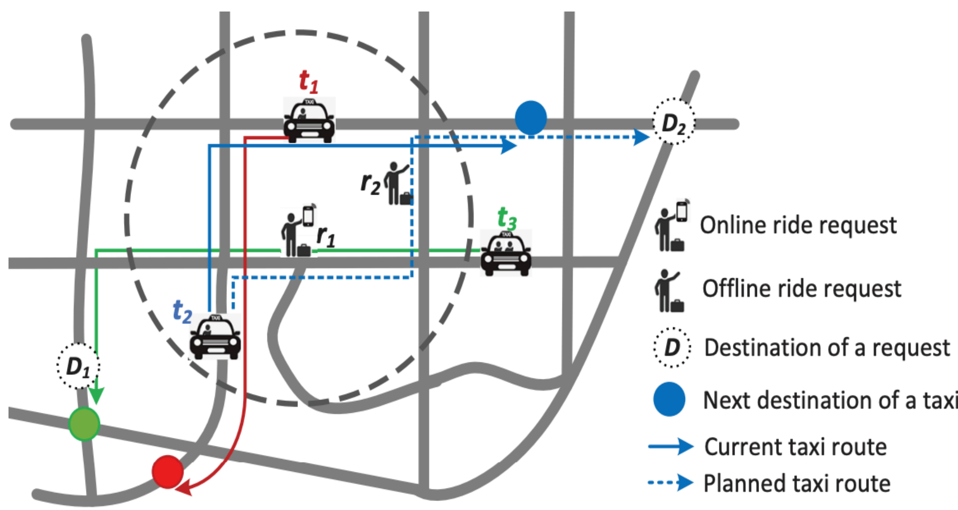
[1] S. Ma, Y. Zheng, O. Wolfson, et al. Real-time city-scale taxi ridesharing. IEEE Transactions on Knowledge and Data Engineering, 27(7):1782– 1795, 2015.

[4] N. Ta, G. Li, T. Zhao, J. Feng, H. Ma, and Z. Gong. An efficient ridesharing framework for maximizing shared route. IEEE Transactions on Knowledge and Data Engineering, 2018.

[5] Y. Xu, Y. Tong, Y. Shi, Q. Tao, K. Xu, and W. Li. An efficient insertion operator in dynamic ridesharing services. In IEEE ICDE, 2019.

Limitations of existing solutions

- Inefficient passenger-taxi matching
  - For serving r_1, t_3 is better than t_1 and t_2
- Omit the offline passengers
  - 55.39% requests like r_2 may be offline [6]

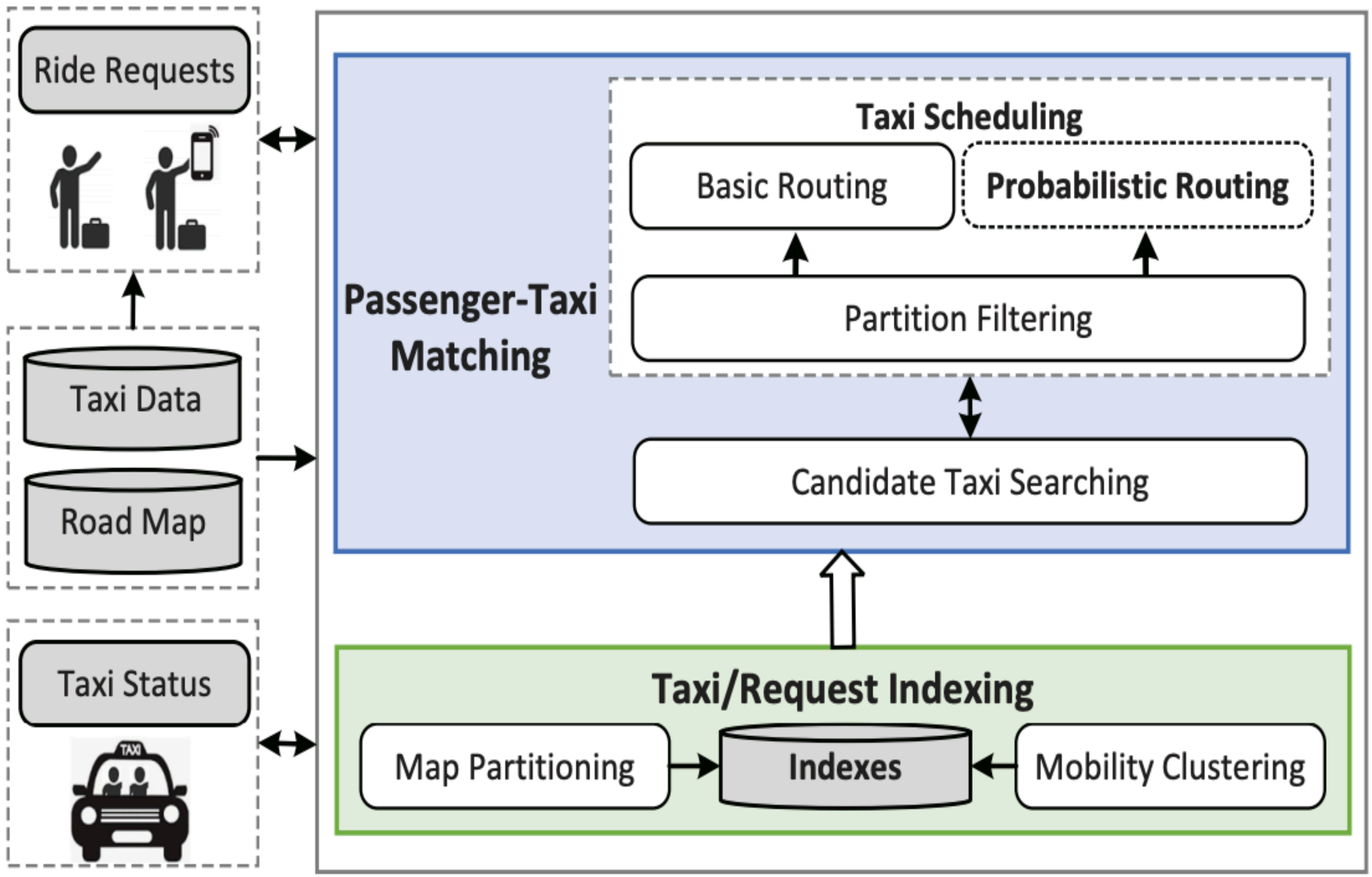


[6] Taxi service research report. http://www.transformcn.com/Topics/ 2018-08/02/b7944fb3-1b99-4840-89d7-eecaaec67bea.pdf.

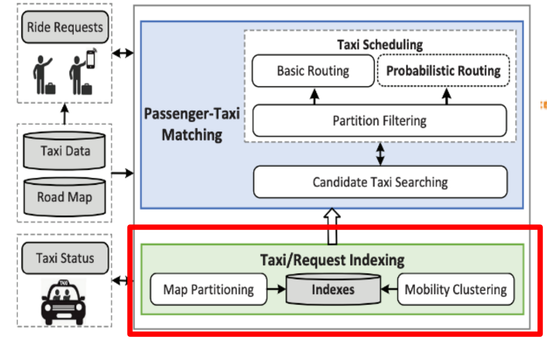
Problem statement

- The mobility-aware taxi ridesharing (MTR) problem
  - Given:
    - A road network ***G***
    - A set of taxis ***T***
    - A set of requests ***R***, including online and offline requests to predict
  - Calculate the schedule/route for each taxi, so as to
    - *Maximize* the number of served ride requests
    - *Minimize* the total detour cost
  - Constraints:
    - Capacity of each taxi
    - Delivery deadline of each request

Our solution: mT-Share



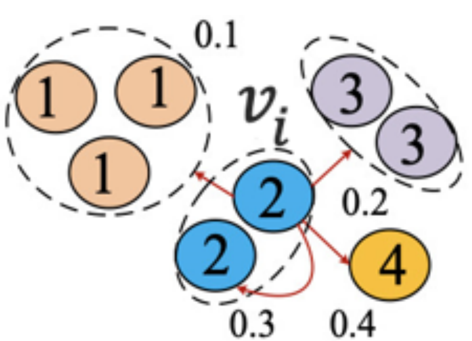
Taxi/request indexing



- Bipartite map partitioning repeats the following 3 steps
  ① Geo-Clustering
  - k-menas based on locations of vertexes

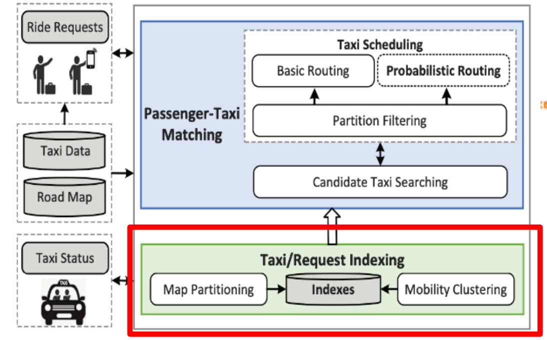
  ② Transition probability calculation
  - Calculated with historical taxi trips

  ③ Transition clustering
  - K-means based on $\overrightarrow{B_i}$ of vertexes



$$\overrightarrow{B_i} = (0.1,0.3,0.2,0.4)$$

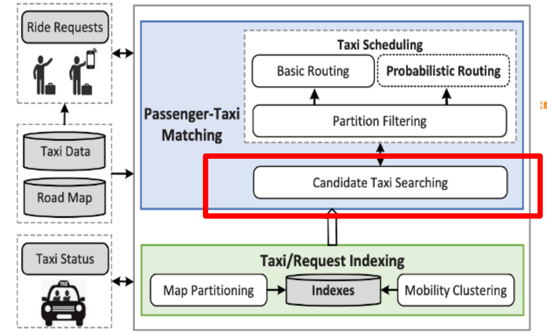
Taxi/request indexing



- Mobility Clustering
  - Mobility Vector
    - Ride request: $\vec{v} = (lng_o, lat_o, lng_d, lat_d)$
    - Cluster with shared request: $\vec{v_c} = \left(lng_o, lat_o, \frac{\sum_{i=1}^{m} lng_{r_i}}{m}, \frac{\sum_{i=1}^{m} lat_{r_i}}{m}\right)$
  - The cosine similarity as the distance metric

$$\cos(\theta) = \frac{\overrightarrow{v_{r_i}} \cdot \overrightarrow{v_C}}{\left|\left|\overrightarrow{v_{r_i}}\right|\right| \times \left|\left|\overrightarrow{v_C}\right|\right|}$$

    - If $\cos(\theta) \geq \lambda$, ride request joins in the mobility cluster C and can share a taxi with other requests of the cluster.



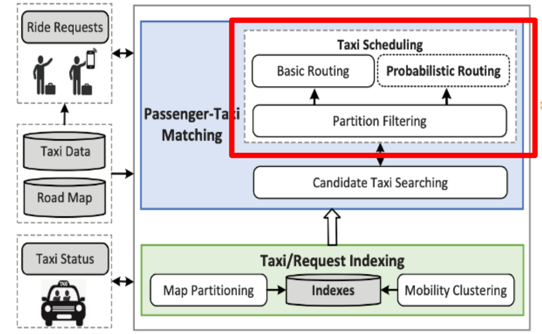
Candidate taxi searching

- Indexes of taxis
  – Map partition based indexing
  – Mobility cluster based indexing

- Candidate taxi searching for a request r
  – Partitions $\mathbb{S}_i$ intersect with a searching radius
  – Mobility cluster C_a to which r belongs

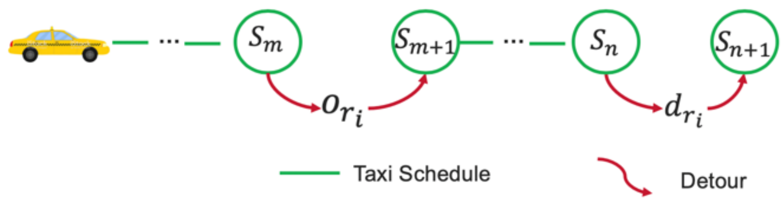
  – Candidate taxi set for request r

$$\mathbb{T}_{r_i} = \{\cup_{P_z \in S_{r_i}} P_z.L_t\} \cap C_a.L_t$$



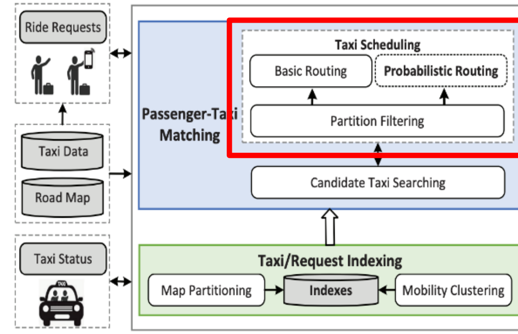
Taxi scheduling

- Select the best taxi to serve r_i
  - Enumerate all possible schedules by inserting o_{r_i} and d_{r_i}



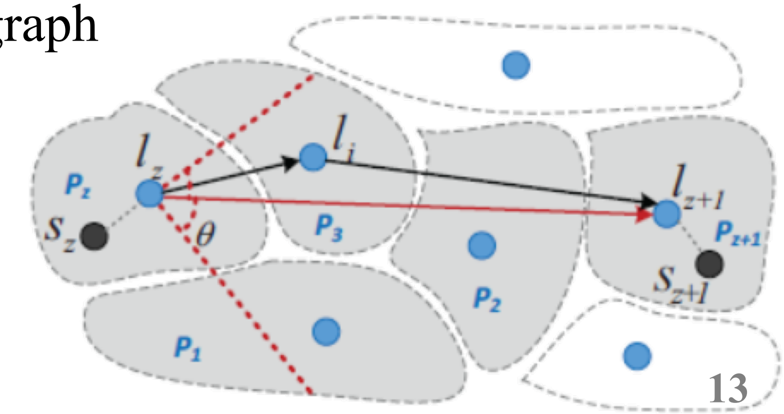
Algorithm 1: Taxi Scheduling

Input: Ride request $\mathrm{r_i}$ and candidate taxi set $\mathbb{T}_{r_i}$;
Output: A taxi with updated schedule/route for $\mathrm{r_i}$;
foreach *taxi* $\mathrm{t_j} \in \mathbb{T}_{r_i}$ **do**
  foreach *schedule instance* $\mathcal{S}'_{t_j} \leftarrow \{\mathcal{S}_{t_j}, o_{r_i}, d_{r_i}\}$ **do**
    if $flag$ **then**
      $\mathcal{R}'_{t_j} = \boxed{ProbabilisticRouting(\mathcal{S}'_{t_j}, \mathrm{t_j});}$
    else
      $\mathcal{R}'_{t_j} = \boxed{BasicRouting(\mathcal{S}'_{t_j}, \mathrm{t_j});}$
    $\omega = cost(\mathcal{R}'_{t_j}) - cost(\mathcal{R}_{t_j});$
Select the taxi schedule instance with the minimum ω;

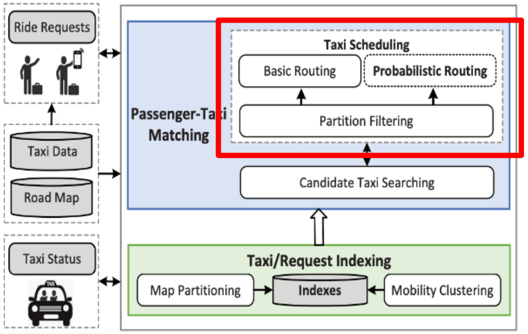


Route planning

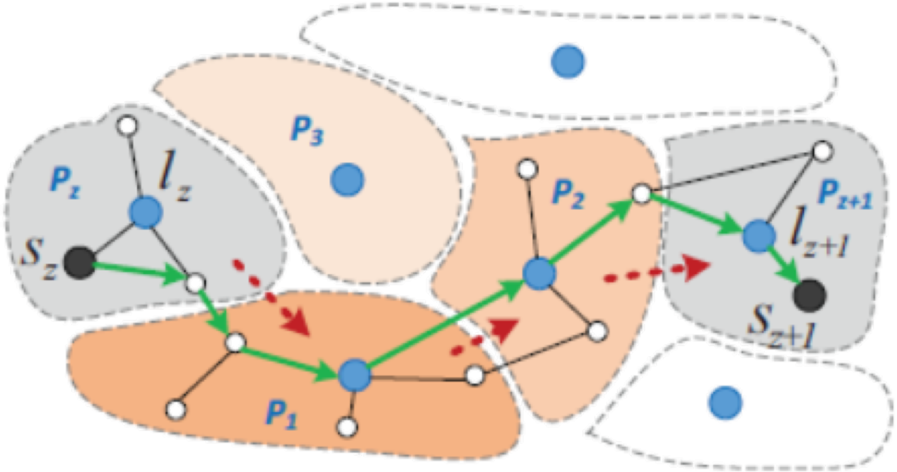
- A two-phase route planning
  - Partition filtering
    - Travel direction rule: $\cos(\theta) = \frac{\vec{v} \cdot \vec{v_z}}{||\vec{v}|| \times ||\vec{v_z}||} \geq \lambda$
    - Travel cost rule: $cost(\mathrm{l}_z, \mathrm{l}_i) + cost(\mathrm{l}_i, \mathrm{l}_{z+1}) \leq (1 + \varepsilon) \times cost(\mathrm{l}_z, \mathrm{l}_{z+1})$
  - Segment-level routing on the filtered partitions
    - Planning route on a reduced subgraph



Segment-level routing



- Basic routing
  - *Dijkstra* -> shortest path
- Probabilistic routing
  - Probability calculation of *suitable passengers*
  - Partition path planning
  - Fine-grained route planning over partition path



Experimental setup

- Dataset
  - 7065907 taxi transactions from Chengdu City, 01/11/2016-31/11/2016, released by Didi GAIA[7]
  - Road Network graph: 214,440 vertices & 466,330 edges
- Simulation scenarios
  - *Peak scenario*: 8:00AM-9:00AM in a workday, with the most requests, i.e., 29,534.
  - *Non-peak scenario*: 10:00AM-11:00AM in a weekend with 15480 requests and 5000 are randomly selected to be the offline requests.

[7] Gaia initiative. https://outreach.didichuxing.com/research/opendata.

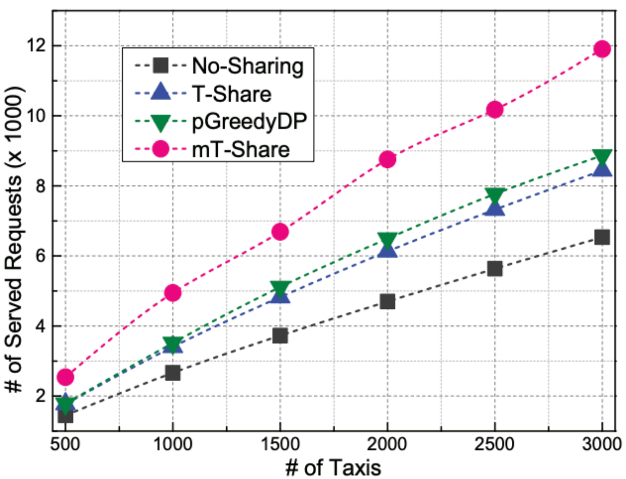
Experimental setup

- Compared schemes
  - No-Sharing
  - T-Share from [1]
  - pGreedyDP [8]
  - mT-Sharepro (enabling the probabilistic routing)
- Performance metrics
  - Number of served requests
  - Response time
  - Detour time

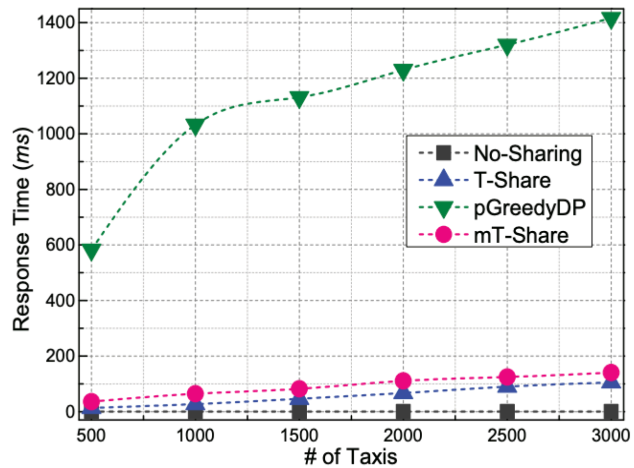
[1] S. Ma, Y. Zheng, O. Wolfson, et al. Real-time city-scale taxi ridesharing. IEEE Transactions on Knowledge and Data Engineering, 27(7):1782– 1795, 2015.

[8] Y. Tong, Y. Zeng, Z. Zhou, L. Chen, J. Ye, and K. Xu. A unified approach to route planning for shared mobility. Proceedings of the VLDB Endowment, 11(11):1633–1646, 2018.

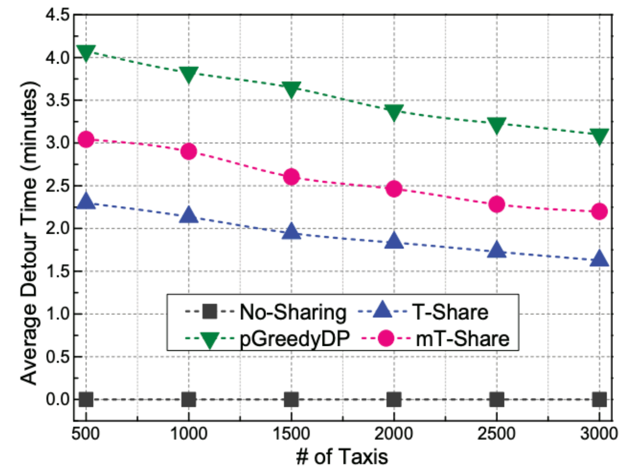
Results in *peak-scenario*



# of Served Requests



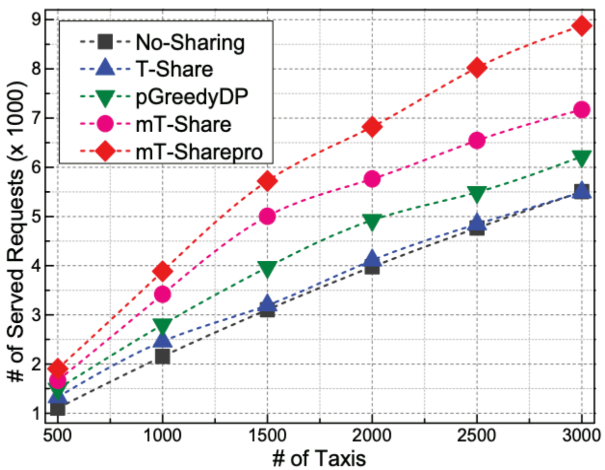
Response time



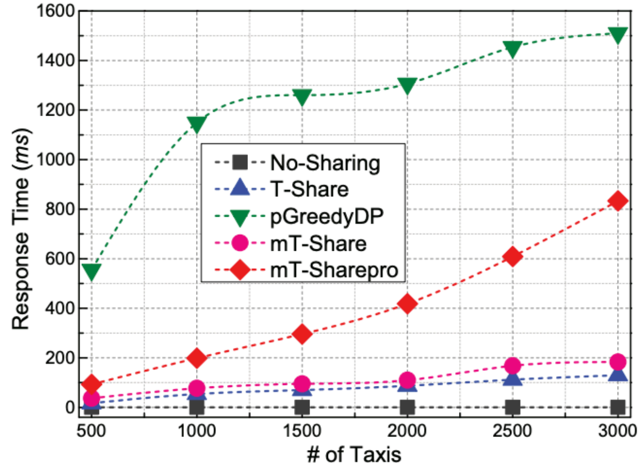
Detour time

mT-Share responses a ride request within 35 ~ 140 ms, and serves 42% and 36% more ride requests than T-Share and pGreedyDP.

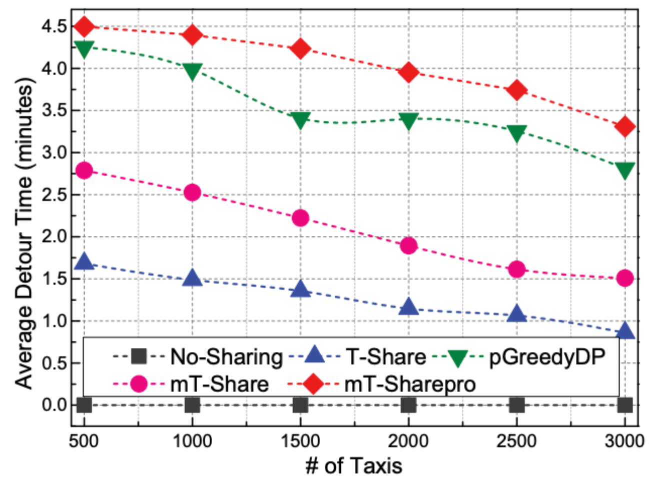
Results in *non-peak scenario*



# of Served Requests



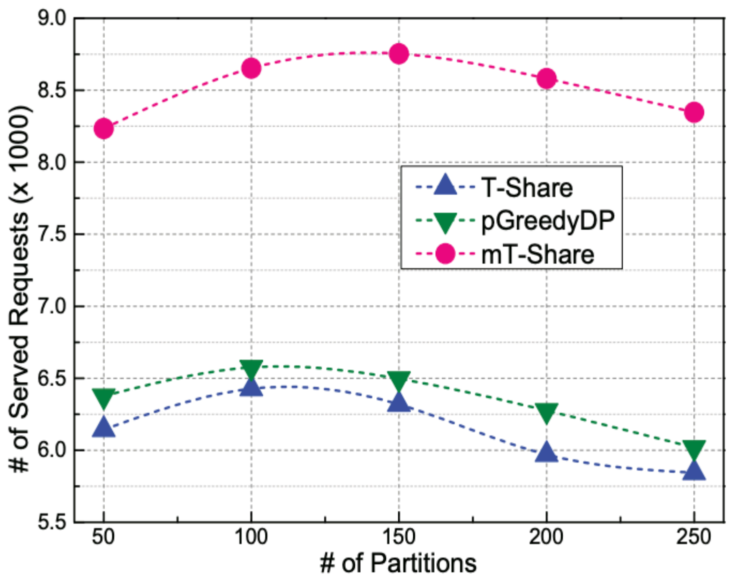
Response time



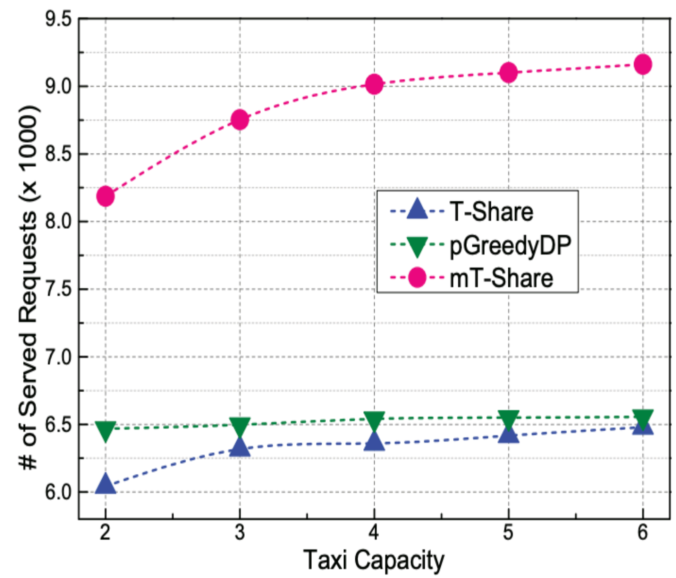
Detour time

Because of the probabilistic routing, mT-Sharepro can server 13% ~ 24% more offline ride requests than T-Share and pGreedyDP.

Experiments on parameter settings



Impact of Partition Number



Impact of Capacity

- ✓ Both too small or oversize partitions will result in a reduced candidate taxi set, and thus affect the ridesharing performance.
- ✓ mT-Share benefits a lot from the larger taxi capacity, while the impact on pGreedyDP is limited.

Conclusion

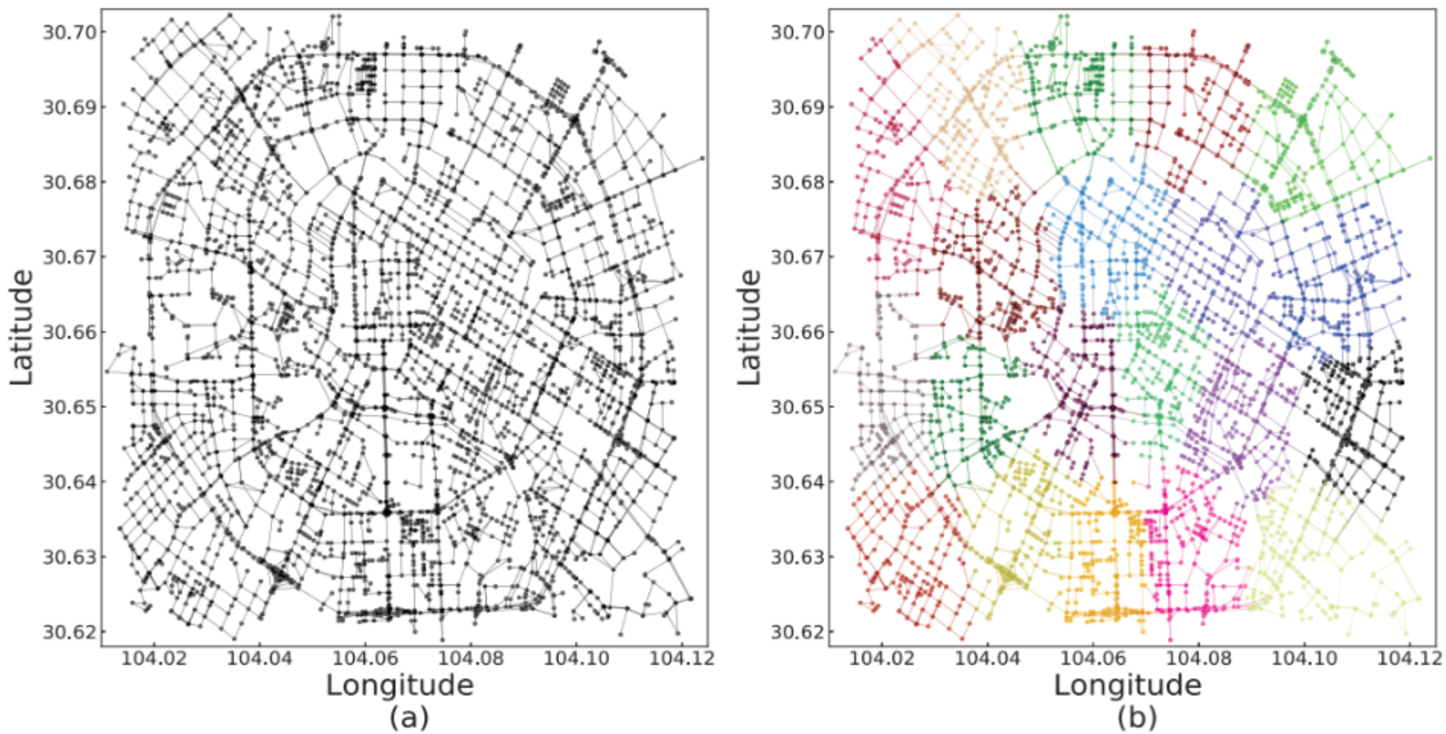
- We consider a novel moility-aware taxi ridesharing problem to serve both offline and online ride requests.
- We propose a novel scheme named mT-Share to improve existing works by fully exploiting the mobility information of taxis and requests.
- Experiments on real-world taxi data show that mT-Share can serve 42% and 62% more ride requests in the peak and non-peak hours than state-of-the-art methods.

Thank You!

Q & A

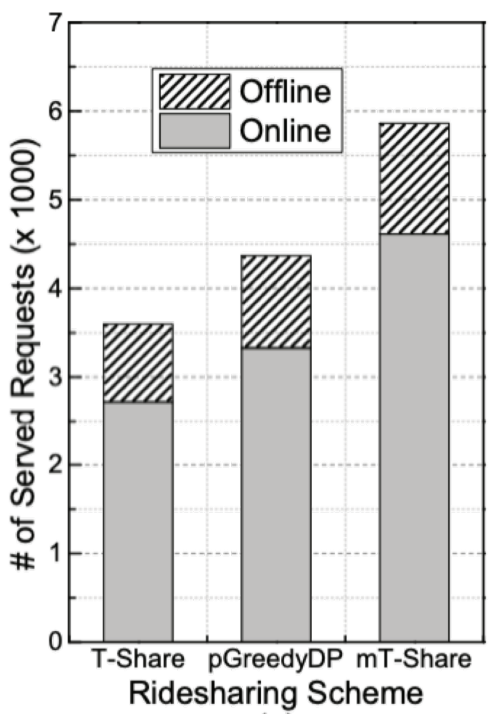
Backup: road network and map partitions

- Road network of Chengdu city from OSM

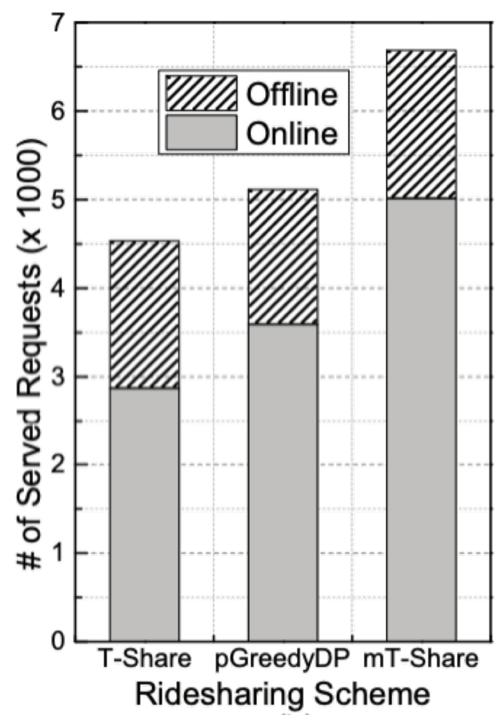


Backup: impact of routing scheme

- Combine basic routing or probabilistic routing with T-Share, pGreedyDP, and mT-Share
  - With probabilistic routing, T-Share, pGreedyDP, and mT-Share can serve 89%, 46%, and 34% more offline ride requests.



(a)



(b)