Optimizing Order Dispatch for Ride-sharing Systems

Yubin Duan , Ning Wang, and Jie Wu

Dept. of Computer and Info. Sciences

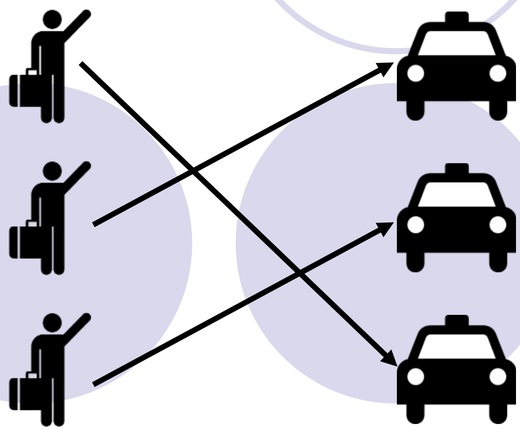
Temple University, USA

Road Map

- Introduction
- Problem Formulation
- Algorithm Design
- Experiment
- Summary

1. Introduction

- Order dispatch in ride-sharing systems
  - passenger: send pickup locations to service provider
  - driver: share real-time locations
  - service provider (SP): dispatch passengers to drivers

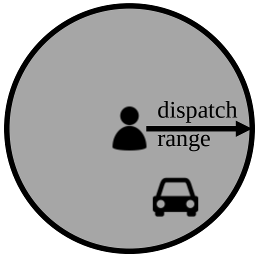


- Existing order dispatch scheme:
  - System-assigning: SP chooses a specific driver for each passenger
  - Driver-grabbing: SP broadcasts passenger locations to drivers

Motivation

- Flaws of existing dispatch scheme:
  - System-assigning:
    - driver preferences [1] are ignored may increase the rejection rate
  - Driver-grabbing:
    - “low- value” orders might take a long time to be accepted

- Combining these two approaches
  - Iteratively enlarge the broadcast region
  - Adaptively set increase ratio based on
    - driver density
    - driver preference (accepting possibility)

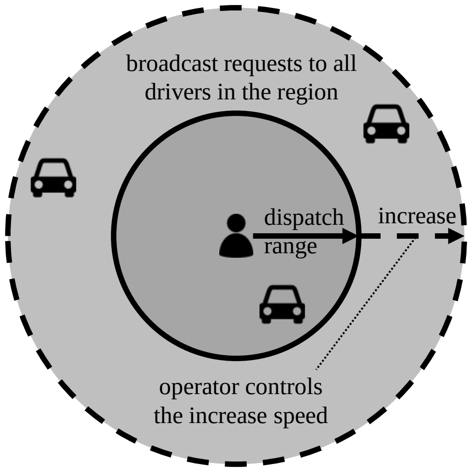


[1] A taxi order dispatch model based on combinatorial optimization (KDD ’17)

Motivation

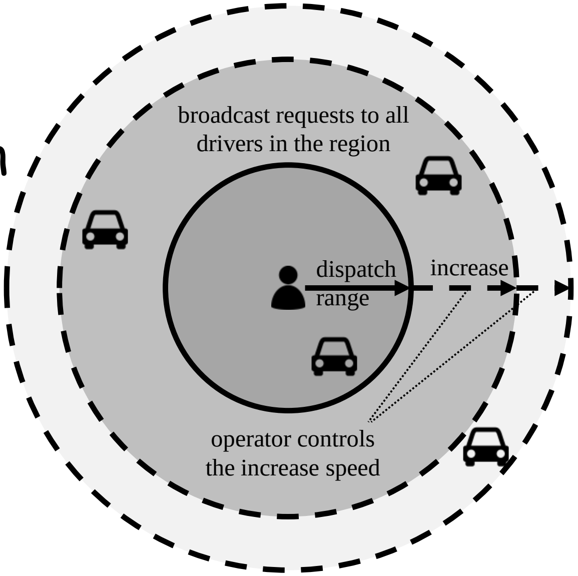
- Flaws of existing dispatch scheme:
  - System-assigning:
    - driver preferences are ignored may increase the rejection rate
  - Driver-grabbing:
    - “low- value” orders might take a long time to be accepted

- Combining these two approaches
  - Iteratively enlarge the broadcast region
  - Adaptively set increase ratio based on
    - driver density
    - driver preference (accepting possibility)



Motivation

- Flaws of existing dispatch scheme:
  - System-assigning:
    - driver preferences are ignored may increase the rejection rate
  - Driver-grabbing:
    - “low- value” orders might take a long time to be accepted
- Combining these two approaches
  - Iteratively enlarge the broadcast region
  - Adaptively set increase ratio based on
    - driver density
    - driver preference (accepting possibility)



Objective

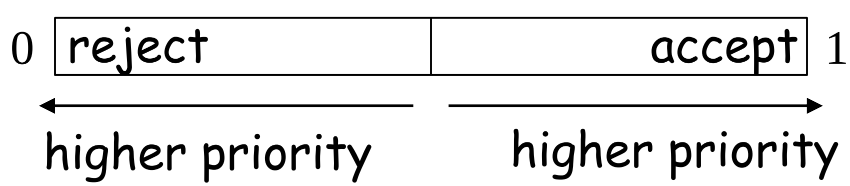
- Joint consider passenger's waiting time and driver's pickup distance

$$\Phi_u = \underbrace{\frac{\mathrm{E}[d_u]}{D_u}}_{\text{pickup distance}} + \alpha \underbrace{\frac{\mathrm{E}[t_u]}{T_u}}_{\text{dispatching time}}$$

- Reducing pickup distance agrees with driver's interest
- Reducing dispatching time agrees with passenger's interest

2. Problem Formulation

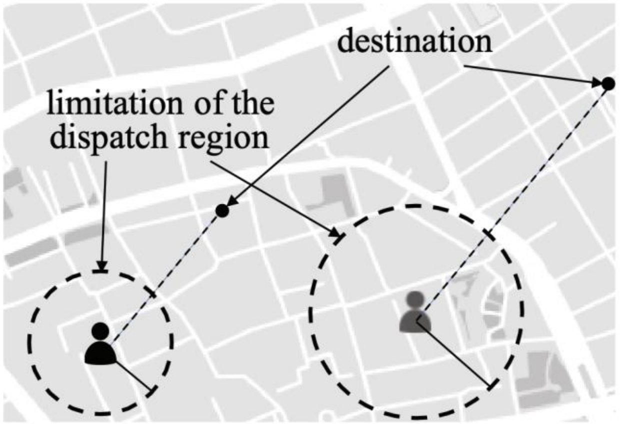
- Pickup preference $p_{u,v}$
  - The probability that driver v accepts the order u
  - Can be learned from history data [1]
- Driver priority is modeled based on p
  - *p=0.5:* hesitate between accepting or rejecting (slower)
  - *p=0* or *1* : certainly reject or accept (faster)
  - Driver priority sorted based on value of |*p-0.5*|



[1] A taxi order dispatch model based on combinatorial optimization (KDD '17)

Probability Model

- Similar as the Geometric distribution
  - The accepting probability for each driver is different
  - The probability of an ordering being accepted $\prod_{i=1}^{k-1}(1-p_{u,i})p_{u,k}$
  - Decision sequence is sorted by driver priority
- Expansion limitation
  - Spatial: the largest region radius is proportional to trip length
  - Temporal: num. of expansions is limited by the longest waiting time



Order Dispatch Problem

- Quantify the objective function
  - The utility function: $\Phi_u = \frac{\mathrm{E}[d_u]}{D_u} + \alpha \frac{\mathrm{E}[t_u]}{T_u}$
  - Expected pickup distance: $\mathrm{E}[d_u] = \sum_{k=1}^{|S_u|} dis(u, v'_k) \prod_{i=1}^{k-1} (1 - p_{u,i}) p_{u,k}$
  - Pickup distance limitation: D_u
- Formulation

$\min \sum \Phi_u$	Minimize utility function
$\sum_{k=1}^{\lvert R_u \rvert} r_{k,u} \leq D_u, \ \forall u \in U$	Dispatch region constraint
$\Delta t \lvert R_u \rvert \leq T_u, \ \forall u \in U$	Waiting time constraint
$r_{k,u} \in \{r \mid r = m\delta, m \in \mathbb{N}\},$	Step length constraint
$1 \leq k \leq \lvert R_u \rvert, \forall u \in U$	

3. Algorithm Design

- Non-overlapping scenario
  - Dispatch regions of different passengers would not overlap
  - A Dynamic Programming Solution
    - state: $f[i][j]$
    - state transfer function

$$f[i][j]=\min_{1\le i\le D,1\le j\le T}\{\underbrace{f[i-1][j-k]}_{\text{previous state}}+\underbrace{\varphi(j-k,j)}_{\text{cost of expanding}},\forall 0\le k\le j\}$$

  - Time complexity: $O(M^2n^3)$, where $n = \max\{D,T\}$

Example

- For non-overlapping scenario

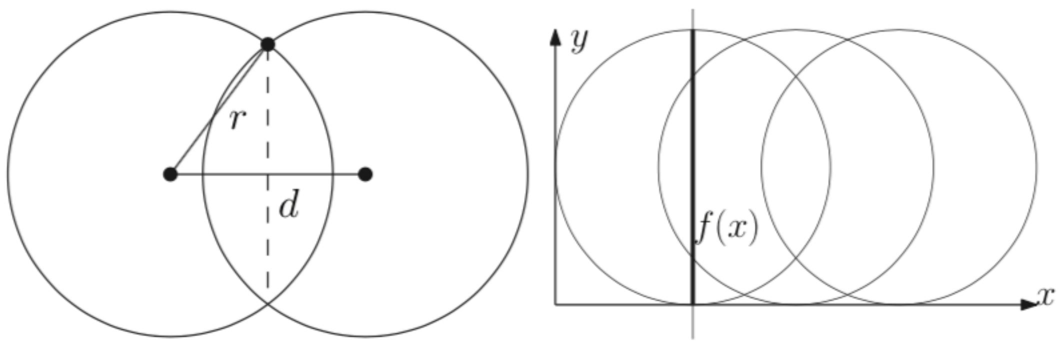
Driver #	1	2	3
Distance to user	0.5	1.5	2.5
Probability to accept the order	0.7	0.9	0.8

- one passenger request
- at most expand 3 times due to time limitation
- spatial step size is set as 1 unit

Expand ratio (# units/iter.)	Utility function value
1, 2, 0	0.486
1, 1, 1	0.490
2, 1, 0	0.651
3, 0, 0	…

Overlapping scenario

- Overlapping scenario for multiple passengers
  - The impact of overlapping

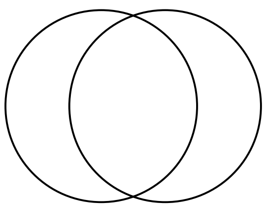


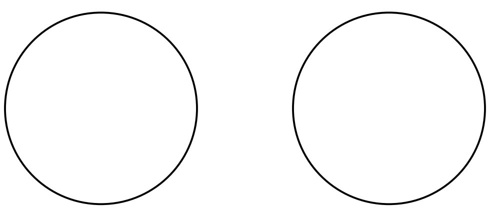
(a) The two-passenger case (b) The general case

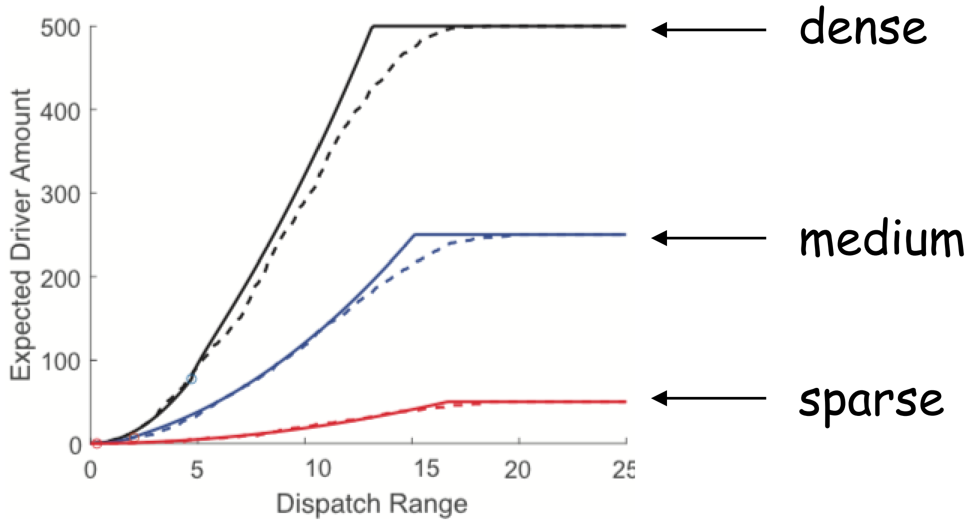
  - Overlapping reduces driver density
    - size of overlapping can be calculated
    - for two passengers: $(2\pi - \arccos \frac{d}{2r})r^2 + d\sqrt{r^2 - \frac{d^2}{4}}$ (Geometric)
    - for more general case: $\int_x^{x+\Delta x} f(x)\mathrm{d}x \approx \frac{\Delta x}{6}\left[f(x)+4f\left(\frac{2x+\Delta x}{2}\right)+f(x+\Delta x)\right]$ (Calculus)

Impact of overlapping

- Visualize the impact
  - dash lines: overlapping case
  - solid lines: non-overlapping case







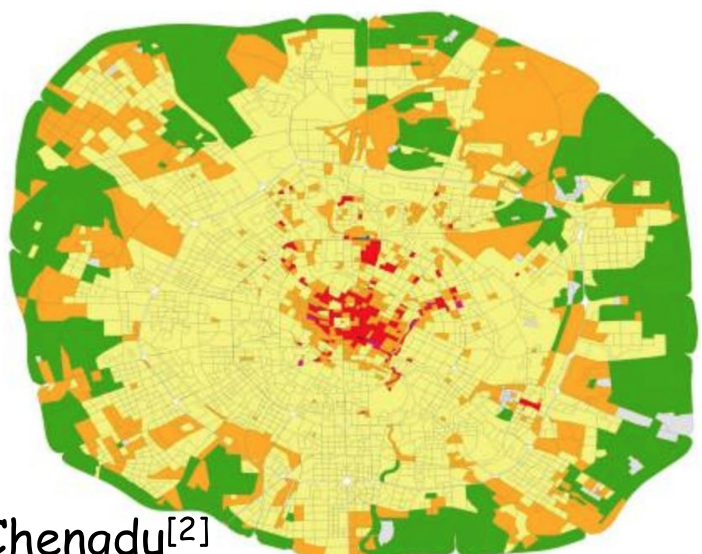
  - more obvious on dense case

4. Experiment

- The DIDI Dataset

Data Source	Didi's trajectory data in Chengdu City
Time Span	11/1/2016 - 11/30/2016
Number of orders	150,000
Average travel distance	8.43 km

- Pickup request distribution



Chengdu[2]

[2] Identification of urban regions' functions in Chengdu, China, based on vehicle trajectory data (NCBI)

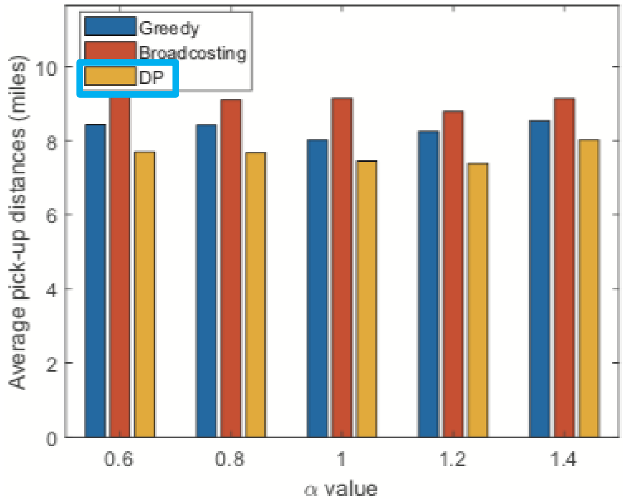
Experiment Setup

- Comparison algorithms
  - Greedy: assigned orders to nearest driver
  - Broadcasting: broadcast orders in the maximum region
  - DP: our algorithm

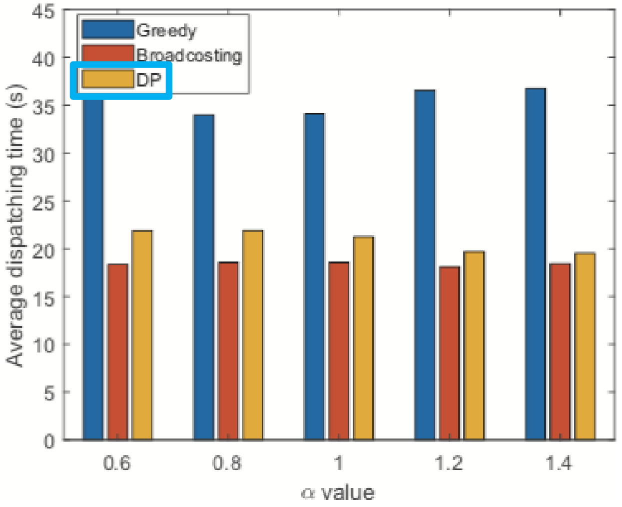
- Settings:
  - The passenger requests are extracted from the Didi dataset
  - Drivers' preferences is learned by the predictor
  - α in the utility function varies from 0.6 to 1.4.

Performance comparison

- On sparse distribution dataset
  - the ratio between the number of divers and the number of passengers is 5
- DP could balance pickup distance and time



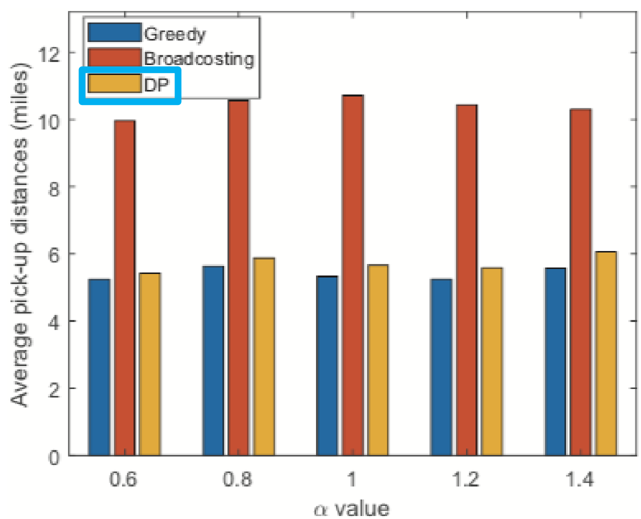
(a) Pickup distance



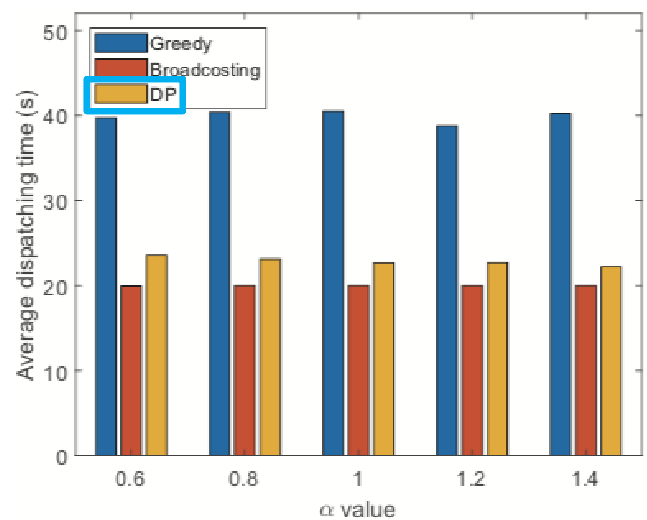
(b) Dispatching time

Performance comparison

- On dense distribution dataset
  - the ratio between the number of divers and the number of passengers is 15
- Similarly, DP could balance pickup distance and time



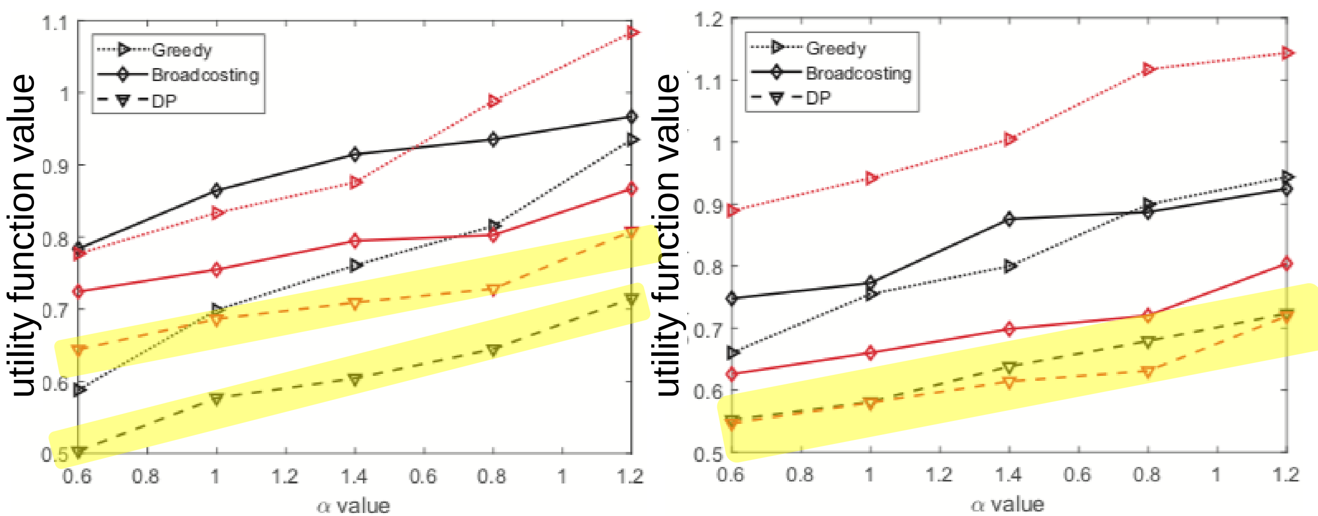
(a) Pickup distance



(b) Dispatching time

Performance comparison

- Comparison on the utility function
  - In both synthetic and real-world dataset, DP could achieve the largest utility function value
  - Red lines: sparse distribution dataset
  - Black lines: dense distribution dataset



(a) On the synthetic dataset (b) On the Didi dataset

5. Summary

- Mixture order dispatch scheme
  - balancing drivers pickup distance and passengers waiting time
- Order dispatch problem
  - maximize the utility function
- Algorithmic solution
  - A dynamic programming algorithm for non-overlapping case
  - Investigate the impact of overlapping
- Experiments on synthetic and real-world dataset
  - Evaluate the performance in terms of the utility function value

Thank you

Q & A