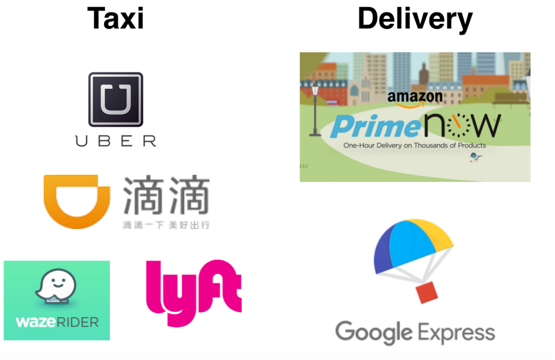
An Optimization Framework For Online Ride-sharing Markets

Yongzheng Jia[1], Wei Xu[1], and Xue Liu[2]

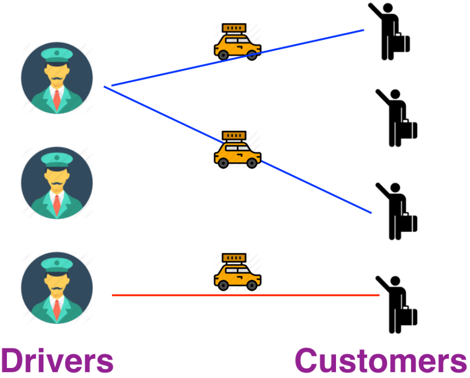
[1]Institute of Interdisciplinary Information Sciences, Tsinghua University
[2]School of Computer Science, McGill University

37^{th} ICDCS – June 7, 2017

- Emergence and rapid development of online ride-sharing services

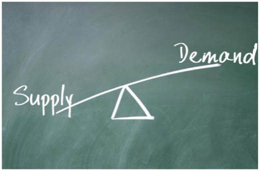


- Two-sided market configuration ⇒ Drivers and Customers
- Existing algorithms are mostly offline heuristics to apply in one-sided market



Efficiency of the services are limited by the sub-optimal and imbalanced matching

- Imbalance between supply and demand (e.g. No match or congestion)
- Long waiting time ⇒ Real-time response
- High cost ⇒ Surge Pricing



- Scalability: Deal with a large number of workers and customers, can partition the map in city's scale (i.e. travel across the entire city)
- Real-time: Always need the platform to give real-time responses to the customers ⇒ Making online algorithms essential

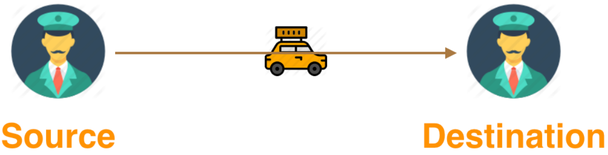
- Generalized economic models for both Internet taxi and product delivery markets
- A deterministic approximation algorithm with a tight theoretical bound
- Two heuristic online algorithms
- Verify the algorithms with theoretical analysis and trace-driven simulations

Two-sided Market with both Temporal and Spatial information

- Drivers - The users who provide taxi or delivery services
- Customers - The users who receive the services
- Tasks - The taxi and delivery services ordered by the customers
- Task Maps - DAGs to demonstrate the relationship between the drivers and tasks in the market

# of drivers: N, and for each driver $n \in [N]$:

- Source - location: $s_n = (u_n^-, v_n^-)$, time: t_n^-
- Destination - location: $d_n = (u_n^+, v_n^+)$, time: t_n^+



# of tasks: M, and for each task $m \in [M]$:

- Source - location: $\bar{s}_m = (\bar{u}_m^-, \bar{v}_m^-)$, time: $\bar{t}_m^-$
- Destination - location: $\bar{d}_m = (\bar{u}_m^+, \bar{v}_m^+)$, time: t_n^+
- Price - p_m (calculated by the platform)
- Publishing time - $\bar{t}_m$: $\bar{t}_m < \bar{t}_m^- < \bar{t}_m^+$

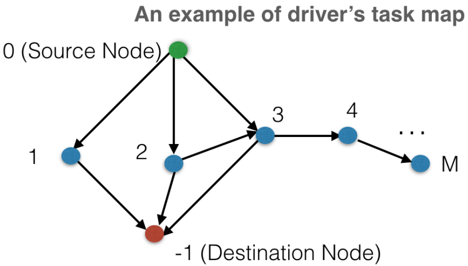


Figure: shows a simple example task map of driver n. The driver can take one task among task 1, task 2 and task 3. She can also take two tasks, and that is to take task 3 after finishing task 2.

Indicator $h_{n,m,m'} \in \{0,1\}, \forall n \in [N], m, m' \in [\hat{M}]$ denotes whether there is an arc from m to m' in driver n's task map.

- $\hat{l}_{n,m}$, $\hat{c}_{n,m}$ - travel time/cost of the same task m for driver n
- $l_{n,m,m'}$, $c_{n,m,m'}$ - travel time/cost of driving empty from m to m' for driver n
- $\hat{h}_{n,m}$ - whether driver n can take task m, with $\hat{h}_{n,m} = 1$ indicating a "yes" as follows:

$$\hat{h}_{n,m} = 1 \Leftrightarrow (\hat{l}_{n,m} \leq \bar{t}_m^{+} - t_m^{-}), \forall n \in [N], m \in [M]. \quad (1)$$

Problem Model
Task Map Construction

For the arcs from the source (labeled 0) to any task m,

$$\begin{aligned} h_{n,0,m} = 1 \Leftrightarrow \hat{h}_{n,m} \wedge (l_{n,0,m} \leq \bar{t}_m^- - t_n^-) \\ \wedge (l_{n,m,-1} \leq t_n^+ - \bar{t}_m^+), \quad \forall n \in [N], m \in [M]. \end{aligned} \tag{2}$$

Tsinghua University

For the arc from one node of task m to the next task m', driver n should have enough time to travel from the destination of task m to the source of task m':

$$h_{n,m,m'} = 1 \Leftrightarrow \hat{h}_{n,m} \wedge \hat{h}_{n,m'} \wedge (l_{n,m',-1} \leq t_n^+ - \bar{t}_{m'}^+) \wedge (l_{n,m,m'} \leq \bar{t}_{m'}^- - \bar{t}_m^+), \forall n \in [N], m \in [M], m' \in [M]. \quad (3)$$

If $h_{n,m,m'} = 1$ then also set $h_{n,m',-1} = 1$, there is an arc from m to m' and another arc from m' to -1.

It will take $(M^2 + 2M)$ iterations to calculate all the values of $h_{n,m,m'}$ for driver $n \Rightarrow$ Complexity to construct the task map of all the N drivers is $O(NM^2)$.

Tsinghua University

Our Goal: Maximize drivers' total profits ⇒ Total Revenue - Total Excess Cost (Shown in (4))

Decision variables:

- $x_{n,m}$ - If task m is assigned to driver n in the market
- $y_{n,m,m'}$ - If driver n takes task m' after finishing task m.

$$Z \text{ :maximize} \sum_{n\in[N]} \sum_{m\in[M]} x_{n,m} p_m - \Big(\sum_{n\in[N]} \sum_{m\in[M]} x_{n,m} \hat{c}_{n,m} + \sum_{n\in[N]} \sum_{m\in[\hat{M}]} \sum_{m'\in[\hat{M}]} y_{n,m,m'} h_{n,m,m'} c_{n,m,m'} - \sum_{n\in[N]} c_{n,0,-1} \Big). \tag{4}$$

Tsinghua University

s.t.

$$\sum_{n\in[N]} x_{n,m} \leq 1, \quad \forall m \in [M]; \tag{5a}$$

$$\sum_{m\in[M]} x_{n,m}p_m \geq \sum_{m\in[\hat{M}]} \sum_{m'\in[\hat{M}]} y_{n,m,m'}h_{n,m,m'}c_{n,m,m'} + \sum_{m\in[M]} x_{n,m} - c_{n,0,-1}, \forall n \in [N]; \tag{5b}$$

$$\sum_{m'\in[\hat{M}]} h_{n,0,m'}y_{n,0,m'} = 1, \quad \forall n \in [N]; \tag{5c}$$

$$\sum_{m\in[\hat{M}]} h_{n,m,-1}y_{n,m,-1} = 1, \quad \forall n \in [N]; \tag{5d}$$

(5a): task allocation, (5b): individual rationality

(5c)-(5d): flow conservation for sources and destinations

$$\sum_{m \in [\hat{M}]} h_{n,m,m'} y_{n,m,m'} = x_{n,m'}, \forall n \in [N], m' \in [M]; \quad (6a)$$

$$\sum_{m' \in [\hat{M}]} h_{n,m,m'} y_{n,m,m'} = x_{n,m}, \forall n \in [N], m \in [M]; \quad (6b)$$

$$x_{n,m} \in \{0,1\}, \quad \forall n \in [N], m \in [M]; \quad (6c)$$

$$y_{n,m,m'} \in \{0,1\}, \quad \forall n \in [N], m \in [\hat{M}], m' \in [\hat{M}]. \quad (6d)$$

(6a)-(6b): flow conservation for internal nodes
(6c) - (6d): decision variables

Tsinghua University

b_m: Customers' willingness-to-pay for task m

$$\hat{Z} : \text{maximize} \sum_{n\in[N]} \sum_{m\in[M]} x_{n,m} b_m - \Big(\sum_{n\in[N]} \sum_{m\in[M]} x_{n,m} \hat{c}_{n,m} + \sum_{n\in[N]} \sum_{m\in[\hat{M}]} \sum_{m'\in[\hat{M}]} y_{n,m,m'} h_{n,m,m'} c_{n,m,m'} - \sum_{n\in[N]} c_{n,0,-1} \Big). \tag{7}$$

s.t

Previous Constrains +

$$\sum_{n\in[N]} x_{n,m}(b_m - p_m) \geq 0, \forall m \in [M]. \tag{8}$$

- Solving (4) or (7) is NP-hard
- In the real markets, it is hard to formulate the social welfare, since it is hard to estimate b_m
- Optimizing the drivers' total profits is enough to improve the efficiency of the ride-sharing markets
- Relax to LP and get an upper bound of OPT

- Original Problem: Allocate tasks to drivers for total profits maximization (temporal + spatial)
- Merge all the N task maps into one DAG (G). Assign each task to at most one driver (Node-disjoint needed).
- Objective: Find multiple *weighted node-disjoint paths* with maximum total value.
- EDP: Edge-disjoint paths (existing solutions)

Tsinghua University

Definitions:

- π: A path from a source to a destination
- $\mathcal{P}_i$: All the paths in the graph G from s_i to d_i for driver i
- f_π:Whether path π is selected in the solution
- r_π: Profit of the path - the summation of the total value of the tasks subtracting the excess cost (defined in Eq. (4))

Tsinghua University

$$Z: \text{maximize} \sum_{\pi \in \cup_i \mathcal{P}_i} f_\pi r_\pi. \tag{9}$$

s.t.

$$\sum_{\pi \in \mathcal{P}_i} f_\pi = x_i, \forall i \in [N]; \tag{10a}$$

$$\sum_{i=1}^{N} \sum_{\pi \in \mathcal{P}_i : m \in \pi} f_\pi \le 1, \forall m \in [M]; \tag{10b}$$

$$x_i \in \{0,1\} \forall i \in [N]; \tag{10c}$$

$$f_\pi \in \{0,1\}, \forall \pi \in \cup_i \mathcal{P}_i. \tag{10d}$$

(9): Same as (4), for the drivers' total profits

(10a): Each driver may choose 1 or 0 task list

(10b): Node-disjoint guarantee

Initialization: Let $S = \emptyset$, $\Pi = \emptyset$, $X = \{1, 2, \cdots, N\}$, $G' = G$

while *there exists driver* $i \in X$ *and path* $\pi \in \cup_i \mathcal{P}_i$ *from* s_i *to* d_i *with strictly positive profit* $r_\pi > 0$ **do**

(a) Find the path $\pi^* = argmax_{\pi \in \cup_i \mathcal{P}_i} r_\pi$, such that π^* has the maximum profit in the current graph G'. Let π^* be the task list for driver i^*;

(b) Remove the source and destination nodes (s_{i^*}, d_{i^*}) of driver i^* and all the task nodes in π^* from the current graph G';

(c) $S = S \cup i^*$, $\Pi = \Pi \cup \pi^*$, $X = X / i^*$;

end

Output the drivers in set S and the selected paths (i.e. task lists) in Π.

Tsinghua University

Theorem

The Greedy Algorithm (i.e. GA) gives a feasible solution with $(\frac{1}{D+1})$-approximation ratio in polynomial time, where D is the maximum number of nodes in a path (i.e. the diameter of the graph G). The ratio is tight.

Tsinghua University

Lemma 1: *Complexity*

GA achieves a feasible solution of (4) within time complexity $O(N^2 M^2)$.

Tsinghua University

Lemma 2: *Lower Bound*

GA guarantees an approximation ratio of $(\frac{1}{D+1})$.

- $\mathcal{B}$: Set of paths selected by *GA*
- $\mathcal{O}$: Paths selected by the optimal solution (i.e. OPT)
- *GA* terminates in K iterations, $\{\pi_k\}_{k=1,2,\cdots,K}$ is the path selected by *GA* during the k-th iteration.

Proposition 1

Every path in $\mathcal{O}$ must intersect with at least one path in $\mathcal{B}$.

Proposition 2

Every path in $\mathcal{B}$ intersects with at most $(D+1)$ paths in $\mathcal{O}$.

$\mathcal{O}_k$: Set of paths in $\mathcal{O}$ that intersect with π_k

Proposition 3

$$\mathcal{O} = \cup_{k=1}^{K} \mathcal{O}_k \tag{11}$$

Proposition 4

$$\sum_{\pi \in \mathcal{O}_k} r_\pi \leq (D+1) \cdot r_{\pi_k}, \quad \forall k = 1, 2, \cdots, K \tag{12}$$

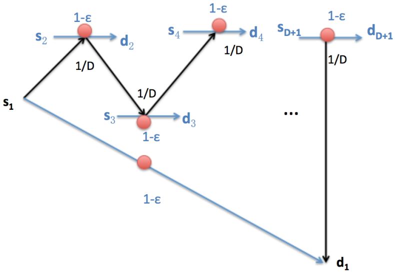
Tsinghua University

Tsinghua University

Lemma 3: *Upper Bound*

$(\frac{1}{D+1})$ is also the upper bound to the approximation ratio.

- $\mathcal{O}$ chooses Blue Edges $\Rightarrow (D+1)\cdot(1-\epsilon)$ (*OPT*)
- $\mathcal{B}$ *chooses* Black Edges $\Rightarrow 1$ (*GA*)



Tsinghua University

- Motivated by the EDP model, state-of-the-art bound: $O\Big(min(n^{2/3}, \sqrt{m})\Big)$ for undirected graphs and $O\Big(min(n^{4/5}, \sqrt{m})\Big)$ for directed graph.
- $(\frac{1}{D+1})$ is a tight bound, and can apply well in real markets. D is small for ride-sharing. $D = 1$ and $\frac{1}{2}$ approximation ratio for Google's Waze Rider market.

Tsinghua University

- When a task m arrives, chooses the driver who can arrive at the first time
- Update the information of tasks and drivers
- If no driver can take the task, then drop task m

- Define the marginal value:
  $\delta_{n,m} = p_m - (c_{n,m,-1} + \hat{c}_{n,m} + c_{n,m',m} - c_{n,m',-1})$
- When a task m arrives, chooses the driver n who can serve with the largest $\delta_{n,m}$
- Update the information of tasks and drivers
- If no driver can take the task, then drop task m

- Dataset: ECML/PKDD 15 including a complete year (from 01/07/2013 to 30/06/2014) of the trajectories for all the 442 taxis running in the city of Porto, Portugal

- $1,000,000+$ records with detailed information, including the timestamp of starting time and finishing time for each trip, polyline of the trip trajectory, and the driver ID

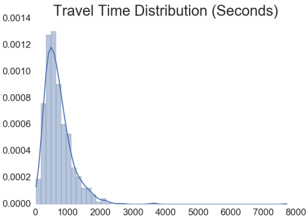


Figure: Travel Time Distribution

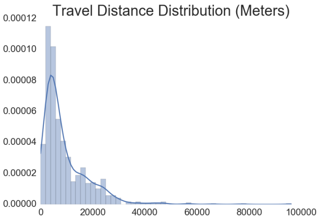


Figure: Travel Distance Distribution

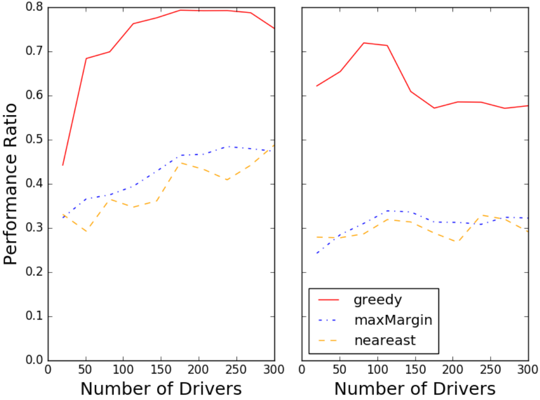
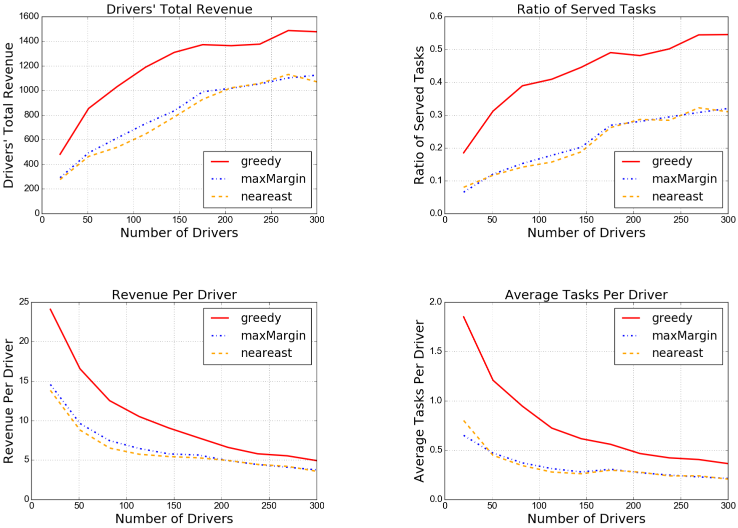


Figure: The left figure shows the performance ratio of the "hitchhiking" model and the right figure shows the performance ratio of the "home-work-home" model

Performance Evaluations

Results: More Insights

Tsinghua University



Thank You

Tsinghua University

Conclusion Remarks

- Propose generalized economic models for ride-sharing markets: Dynamic Scheduling based on Temporal + Spatial info
- A deterministic offline algorithm + Two online heuristics
- Application Specialization: Limited # of tasks within a period, our greedy algorithm works fine
- Future Work: Design deterministic online algorithms

Contact Information - Yongzheng Jia

- jiayz13@mails.tsinghua.edu.cn

Thank You

Yongzheng Jia
jiayz13@mails.tsinghua.edu.cn
Wechat ID: jiayz90