Online to Offline Business: Urban Taxi Dispatching with Passenger-Driver Matching Stability

Huanyang Zheng and Jie Wu
Dept. of Computer and Info. Sciences
Temple University

Road Map

- Introduction & Motivation
- Problem Formulation & Analysis
- Non-sharing taxi dispatching
- Sharing taxi dispatching
- Experiments
- Conclusion

Introduction & Motivation

Traditional taxi business

- Taxi belongs to the company
- Driver's interest is ignored during taxi dispatch

Online to offline taxi (e.g., uber)

- Taxi does not belong to the company
- Driver's interest must be considered during taxi dispatch
- Balanced interest between passengers and drivers

Introduction & Motivation

Taxi dispatch (i.e., matching)

- Time-window-based dispatch -> match in snapshot
- Non-sharing: one request -> one taxi
- Sharing: multiple requests -> one taxi

Challenges

- Matching fairness between passengers and drivers
- Complexity and scalability of dispatch

Problem Formulation & Analysis

Taxi dispatch Scenario

- A set of taxis, $T = \{t_i\}$
- A set of passenger request, $R = \{r_j\}$
- r_j^s and r_j^d : pick-up and drop-off locations of r_j
- Distance function, $D(\cdot\,,\cdot)$

Formulation by matching

- Stable marriage approach

Problem Formulation & Analysis

Extended stable marriage (taxi and request)

- # of taxis does not equal to # of requests
- Allow empty/dummy match (i.e., taxi not dispatched)
- Preference list includes empty/dummy entry

Classic stable marriage (man and woman)

- # of men does equal to # of woman
- Does not allow empty/dummy match
- Preference list does not include empty/dummy entry

Problem Formulation & Analysis

Extended stable marriage (taxi and request)

- Each taxi (request) ranks requests (taxis) as its preference list, including an empty/dummy entry

Extended stability

- A matching (i.e., taxi dispatch schedule) is stable, if an arbitrary matched passenger and another arbitrary matched driver will not prefer each other over their existing partners (including dummies partners, and dummies always prefer non-dummies over dummies).

Problem Formulation & Analysis

Extended stable marriage (taxi and request)

Theorem 1: A stable matching exists, if exact one dummy entry exists in the preference order of each passenger request and that of each taxi.

- Proof idea: convert extended stable marriage to traditional stable marriage by adding dummies
- Schedulability for taxi dispatch is guaranteed

Non-sharing Taxi Dispatch

One request -> one taxi

- Two sub-algorithms: proposal and refusal

Proposal sub-algorithm

- Request proposes to taxi (terminated at dummy)

Refusal sub-algorithm

- Taxi accepts proposals that are better than current, and refuses current (otherwise refuse proposal)

Non-sharing Taxi Dispatch

Proposal sub-algorithm

- Request proposes to taxi (terminated at dummy)

Sub-algorithm Proposal
Input: passenger request r_j.
if the next entry in r_j's preference order is non-dummy (let t_i denote this entry) **then**
  Update r_j's current entry to be t_i.
  Call sub-algorithm Refusal for S^*, t_i, and r_j.
else
  r_j is unserved (no taxi dispatch), and update S^*.

Non-sharing Taxi Dispatch

Refusal sub-algorithm

- Taxi accepts proposals that are better than current, and refuses current (otherwise refuse proposal)

Sub-algorithm Refusal
Input: schedule S^*, taxi t_i, and passenger request r_j.
if t_i is undispatched and prefers r_j over no dispatch **then**
    Dispatch t_i to r_j, and update S^*.
else if t_i is dispatched to $r_{j'}$, but prefers r_j over $r_{j'}$ **then**
    Dispatch t_i to r_j, and update S^*.
    Call sub-algorithm Proposal for $r_{j'}$.
else
    Call sub-algorithm Proposal for r_j.

Non-sharing Taxi Dispatch

Non-sharing taxi dispatch

- Request's preference: distance to taxi (waiting time)
- Taxi's preference: distance to taxi (waiting time) and distance from pick-up to drop-off (carry distance)

Non-Sharing Taxi Dispatch

Compute preference orders for $\forall r_i \in R$ and $\forall t_j \in T$ based on $D(t_i, r_j^s)$ and $D(t_i, r_j^s) - \alpha D(r_j^s, r_j^d)$, respectively.

Initialize each taxi in T as undispatched.

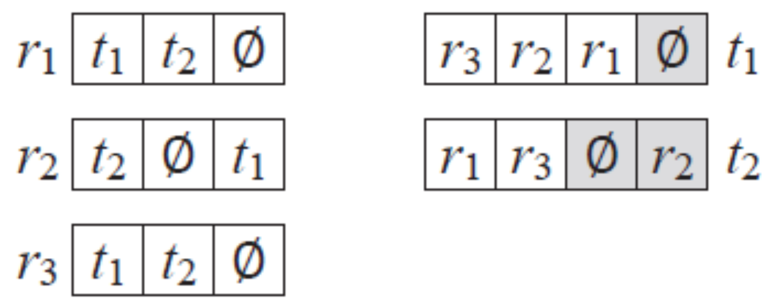
for each passenger request, $r_j \in R$ **do**

  Call sub-algorithm Proposal for r_j.

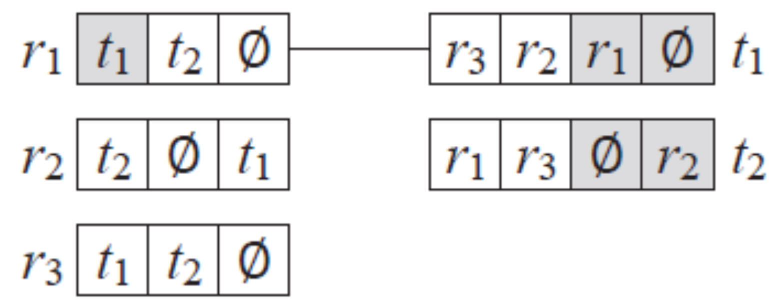
return S^* as a passenger-optimal taxi dispatch schedule.

Non-sharing Taxi Dispatch

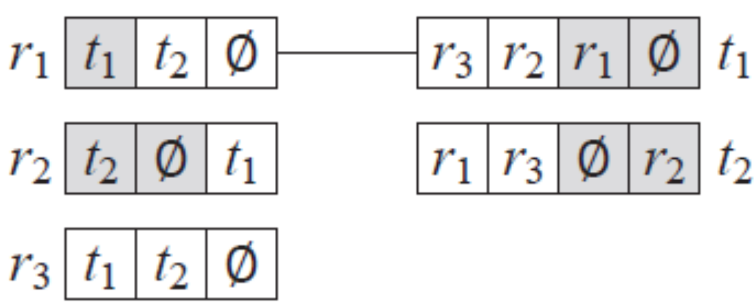
Algorithmic example



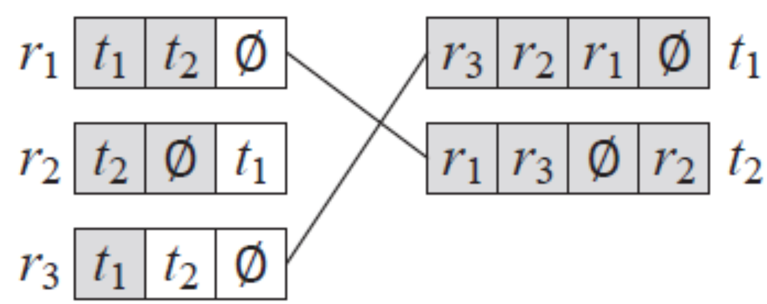
(a) Initialization.



(b) r_1's proposal.



(c) r_2's proposal.



(d) r_3's proposal.

Non-sharing Taxi Dispatch

Property 1: In the stable matching obtained by Algorithm 1, if a taxi prefers no dispatch over all passenger requests, then it will not be dispatched. Similarly, if a passenger request prefers no service over all taxis, then it will not be served.

Property 2: Among all stable matchings, the stable matching obtained by Algorithm 1 satisfies that passenger requests have their best partners, but taxis have their worst partners.

Theorem 2: In the passenger-optimal stable matching obtained by Algorithm 1, if a passenger request is unserved, it is unserved in all possible stable matchings.

Sharing Taxi Dispatch

Multiple requests -> one taxi

Theorem 3: Given a set of request and a taxi, it is NP-hard to compute a directed shortest path that goes through the pick-up location before the drop-off location for each passenger request.

- Reduction from Shortest Hamiltonian Path Problem
- In practice, only pack 2 to 3 requests to a taxi
- Exhaustive search is used to determine route

Sharing Taxi Dispatch

Packing 2 to 3 requests

- Exhaustively compute all possible request sets
- Eliminate bad sets by threshold
- Use minimum set packing to pack requests

$$
\begin{aligned}
\text{maximize} \quad & \sum_{k} x_k \\
\text{subject to} \quad & \sum_{k: r_j \in c_k} x_k \leq 1 \text{ for } \forall j \\
& x_k \in \{0, 1\} \text{ for } \forall k
\end{aligned}
$$

- Taxi's and passenger's interests need to consider sharing

Experiments

Real data-driven

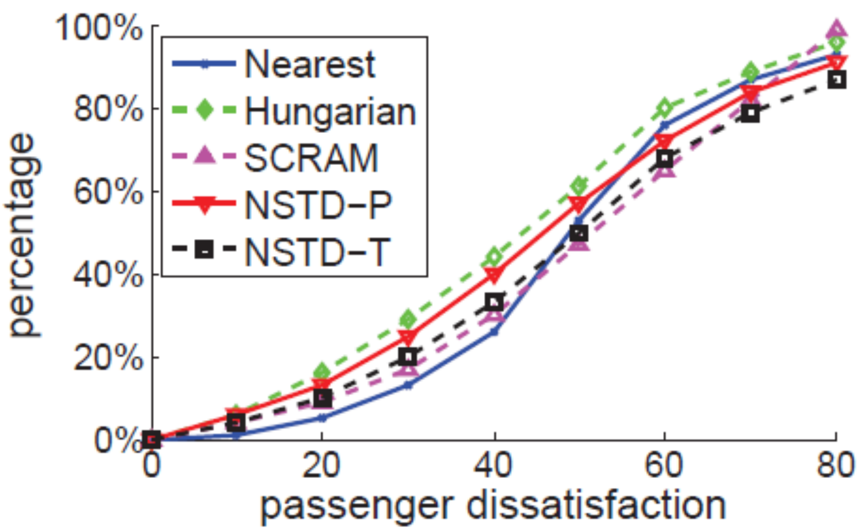
- New York trace: 1,445,285 requests and 700 taxis
- Boston trace: 406,247 requests and 200 taxis
- Taxi speed: set to 20km/h
- Dispatch window: 1 minute

Comparison algorithm

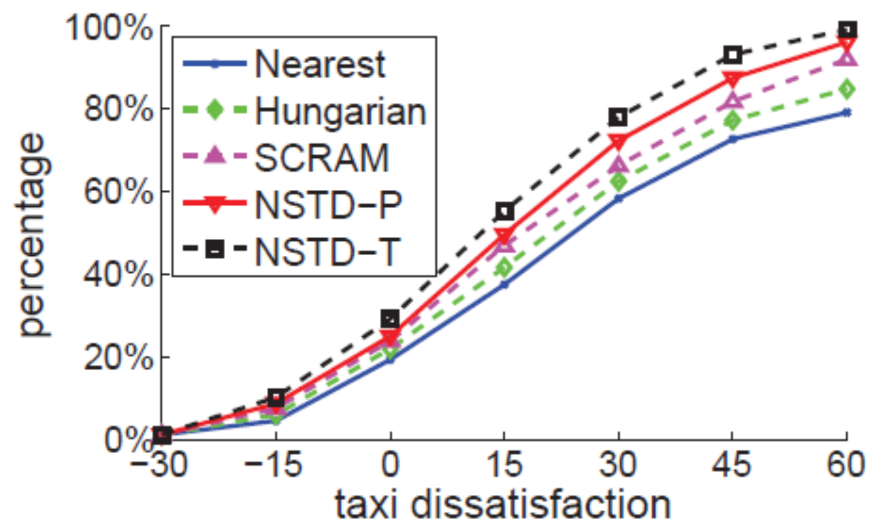
- Nearest, Hungarian, SCRAM (minimize maximum)

Experiments

Non-sharing taxi dispatch (New York trace)



(b) Passenger dissatisfaction distribution.

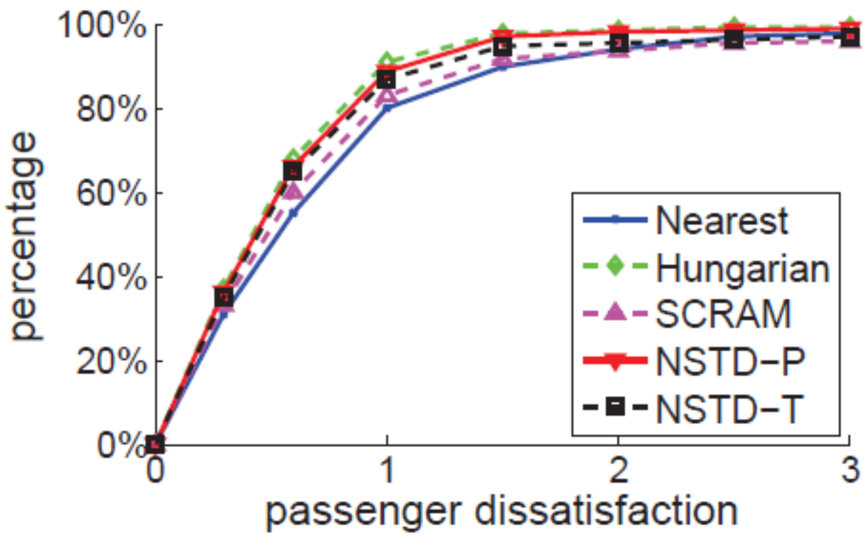


(c) Taxi dissatisfaction distribution.

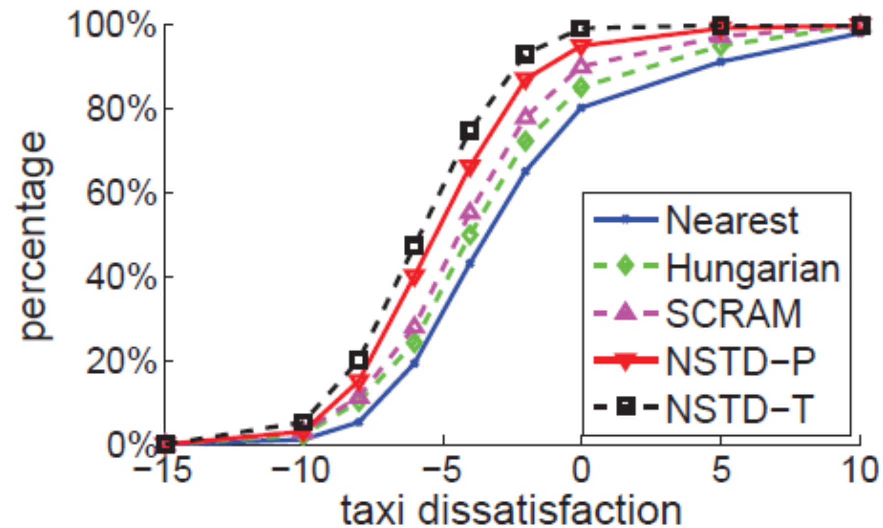
- Passenger's dissatisfaction slightly increases, but taxi's dissatisfaction significantly decreases

Experiments

Non-sharing taxi dispatch (Boston trace)



(b) Passenger dissatisfaction distribution.



(c) Taxi dissatisfaction distribution.

- Passenger's dissatisfaction slightly increases, but taxi's dissatisfaction significantly decreases

Conclusion

Online to offline taxi (e.g., uber)

- Balanced interest between passengers and drivers
- Extended stable marriage approach (enable dummy)

Non-sharing and sharing taxi dispatches

- Taxi's / request's interest (waiting time & carry distance)
- Pack passenger requests to a taxi
- Certain algorithm properties

End

Questions & Answers