Journal of Southeast University (English Edition) Vol. 42, No. 2, pp. 250-256 June 2026 ISSN 1003-7985

# Stable matching mechanism for multi-modal integration on mobility-as-a-service platform

WU Yating, LAI Minghui

(School of Economics and Management, Southeast University, Nanjing 211189, China)

**Abstract**: In mobility-as-a-service (MaaS) platforms integrating ridesharing with public transit, each rider-driver pair may have multiple potential matches via different transfer nodes, with users being self-interested with heterogeneous preferences. After generating all feasible integrated matches, a two-sided one-to-one stable matching model is formulated to maximize platform revenue, where each feasible match corresponds to a stability constraint embedding preference information. To solve this model efficiently, an iterative constraint-generation algorithm is designed. It repeatedly solves a restricted master problem to obtain a temporary solution and a subproblem to identify violated stability constraints, iterating until no violations remain. The proposed algorithm can significantly improve computational efficiency. Compared with a centralized matching benchmark with the blocking rate up to 75%, stable matching increases transit usage and user acceptance at the cost of a 31.43% reduction in average platform revenue. Riders experience longer detours with greater cost savings, whereas drivers exhibit the opposite pattern.

**Key words**: mobility as a service; stable matching; constraint generation; ridesharing; public transit

**DOI**: 10.3969/j.issn.1003-7985.2026.02.012

With advances in information and communication technologies and the rise of the sharing economy, mobility as a service (MaaS) has emerged as a transformative paradigm, shifting mobility from ownership-based to usage-based systems[1]. By integrating multi-modal transport, MaaS platforms offer one-stop, personalized services that enhance the accessibility and appeal of public transit[2]. Implementations such as RideMyRoute in Europe[3] and the Shenzhen MaaS project in China[4] demonstrate the advantages of combining ridesharing with transit via transfer nodes.

Because the selection of transfer nodes is critical for user acceptance[5], we study the operational problem of the MaaS platform integrating peer-to-peer ridesharing with public transit and allowing flexible selection of transfer stations. The platform serves two types of users: riders seeking transportation services and private-car drivers providing compensated rides en route to their own destinations. The task is to match each rider with a driver and/or transit. In this respect, two challenges arise. First, each rider-driver pair may have multiple match options via different transfer nodes, which enlarges the match space and complicates optimization. Second, users are self-interested with heterogeneous preferences[1,6], so assignments that do not raise individual utility may be rejected, undermining participation and platform sustainability[7]. Accordingly, we aim to deliver stable, user-acceptable assignments while preserving operational efficiency, reflecting trade-offs similarly observed in Internet of Vehicles resource allocation[8].

The MaaS literature mostly focuses on conceptual development, governance frameworks, or empirical analyses of adoption barriers and user attitudes[9-10]. Existing quantitative works primarily develop centralized matching models integrating ridesharing with transit[11-12]. However, these models typically assume user compliance, thus ignoring heterogeneous preferences. To fill this gap, we cast the problem as a two-sided one-to-one stable matching based on user preferences.

The two-sided one-to-one stable matching problem[13-14] can be solved using the Gale-Shapley algorithm[13], but this does not optimize platform revenue or social welfare. We therefore adopt a preference-based integer program with complex stability constraints to optimize these objectives[7,15-16]. In the ridesharing context, Wang et al. [7] proved total unimodularity (TUM) under a structure where each rider-driver pair has a single match option. Consequently, linear relaxation yields integral solutions. In our integrated case, each pair has multiple options, violating the structure required for TUM. Linear programming relaxation is therefore nonintegral, and TUM does not hold. Consequently, the integer program with numerous complex stability constraints is computationally challenging in practice.

To solve this problem efficiently, we develop an algorithm based on a constraint-generation procedure. Extensive experiments demonstrate the computational effi-

**Received** 2025-07-07, **Revised** 2025-09-26.

**Biographies**: WU Yating (1998—), female, Ph. D. graduate; LAI Minghui (corresponding author), male, doctor, professor, laimh@seu.edu.cn.

**Foundation items**: The National Natural Science Foundation of China (No. 72231002, 72371070).

**Citation**: WU Yating, LAI Minghui. Stable matching mechanism for multi-modal integration on mobility-as-a-service platform [J]. Journal of Southeast University (English Edition), 2026, 42(2): 250-256. DOI: 10.3969/j.issn.1003-7985.2026.02.012.

ciency of our algorithm. We further find that stable matching increases transit usage, aligning with platform sustainability goals.

## 1 MaaS Matching Problem

We consider a mobility platform operating a MaaS system to provide morning commuting services by integrating public transit and ridesharing. The public transit network is noncongested, reliably operated by a nonprofit agency, with travel times determined by shortest paths[11-12]. The system has two types of users $U = R \cup D$: riders $R$ who seek transportation services and drivers $D$ offering rides toward their destinations for compensation. Each user submits a request specifying the origin $o_i$, destination $d_i$, earliest departure $T_i^-$, latest arrival $T_i^+$, and maximum acceptable duration $T_i^{\max}$. Accordingly, the platform flexibly matches riders with drivers and/or transit. A rider may complete a trip by ridesharing, transit, or both. When ridesharing is involved, the transfer station need not be the nearest.

### 1.1 Integrated matching

To reduce user inconvenience, we only restrict matches to one-to-one rider-driver pairings. The platform offers four integrated travel options, depending on whether a rider's origin or destination is within the maximum walking distance $\bar{W}$ to a transit station. If so, the rider may walk to or from the nearest station; otherwise, ridesharing is used for the first- or last-mile segment. Options requiring ridesharing at both segments are excluded[11,17].

For convenience, public transit is treated as a virtual driver indexed by 0; thus, $\bar{D} = D \cup \{0\}$. We define $V = N \cup \{d_i\}_{i \in R}$ as transfer nodes, including transit stations $N$ and rider destinations. A match is denoted by a triplet $(i, j, v)$, where $i \in R$, $j \in \bar{D}$, and $v \in V$. Let $t_C(\cdot)$, $t_P(\cdot)$, and $t_W(\cdot)$ denote the driving, transit, and walking time, respectively. The following four match types are illustrated in Fig. 1: (1) Sole ridesharing (SR) match. Rider $i \in D$ is directly driven to $d_i$ by driver $j \in D$, denoted as $(i, j, d_i)$. (2) Ridesharing-transit (RT) match. Ridesharing is used for the first-mile segment to a station $v \in N$, denoted as $(i, j, v)$. (3) Transit-ridesharing (TR) match. Ridesharing is used for the last-mile segment from a station $v \in N$, denoted as $(i, j, v)$. (4) Sole transit (ST) match. Rider $i$ travels solely by transit, denoted as $(i, 0, d_i)$.

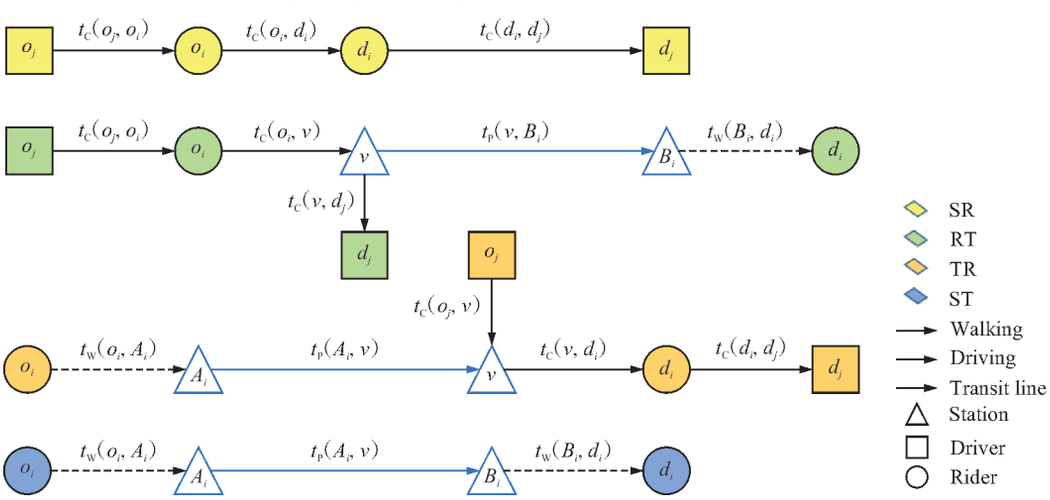


**Fig. 1** Four types of travel modes

### 1.2 Time feasibility

A match $(i, j, v)$ is time-feasible if it satisfies both users' earliest departure, latest arrival, and maximum trip duration constraints. The earliest time the driver can pick up rider $i$ at $o_i$ or $v$ is

$$T_{ijv}^{\text{pick}} = \begin{cases} \max\left\{T_i^-,\ T_j^- + t_C(o_j, o_i)\right\} \\ \qquad\qquad (i, j, v) \text{ is an SR/RT match} \\ \max\left\{T_j^- + t_C(o_j, v),\ T_i^- + t_W(o_i, A_i) + t_P(A_i, v)\right\} \\ \qquad\qquad (i, j, v) \text{ is a TR match} \end{cases}$$

Let $\tau_i(i, j, v)$, $\tau_j(i, j, v)$ denote the total trip durations of rider $i$ and driver $j$, respectively, and $\tau_i^{\text{pick}}(i, j, v)$, $\tau_j^{\text{pick}}(i, j, v)$ represent their durations after pickup. The duration calculations depend on the match type, as shown in Fig. 1. Then, time feasibility requires $T_{ijv}^{\text{pick}} + \tau_i^{\text{pick}}(i, j, v) \leqslant T_i^+$, $T_{ijv}^{\text{pick}} + \tau_j^{\text{pick}}(i, j, v) \leqslant T_j^+$, $\tau_i(i, j, v) \leqslant T_i^{\max}$, and $\tau_j(i, j, v) \leqslant T_j^{\max}$.

### 1.3 Cost feasibility

Users participate only if their generalized travel costs (including monetary and time costs) do not exceed the cost of traveling alone. Let $\theta_i^0$ denote user $i$'s traveling-alone cost. We assume that riders take transit for solo travel only if their nearest access and egress stations $A_i$

and $B_i$ both lie within $\bar{W}$; otherwise, they use ridesourcing. Let $f_C$, $l(\cdot)$, $\alpha$, and $f_i^R$ denote the per-distance driving cost, distance, value of time, and ridesourcing fare, respectively.

$$\theta_i^0 = \begin{cases} f_C l(o_i, d_i) + \alpha t_C(o_i, d_i) & \text{Driver by car} \\ f_P(A_i, B_i) + \alpha[t_W(o_i, A_i) + t_P(A_i, B_i) + t_W(B_i, d_i)] & \text{Rider uses transit} \\ f_i^R + \alpha t_C(o_i, d_i) & \text{Rider uses ridesourcing} \end{cases}$$

Under a match $(i, j, v)$, user $i$'s generalized travel cost is $\theta_i(i, j, v)$. For each driver $j \in D$, $\theta_j(i, j, v) = f_C L_j(i, j, v) + \alpha\tau_j(i, j, v) + (1 - \lambda)p(i, j, v)$, where $L_j(\cdot)$ denotes driver $j$'s total trip distance. The platform applies a predetermined pricing scheme $p(i, j, v)$ (e. g., Dida Chuxing[18]) and charges a commission of $\lambda p(i, j, v)$. For each rider $i \in R$,

$$\theta_i(i, j, v) = \begin{cases} p(i, j, v) + \alpha\tau_i(i, j, v) & (i, j, v) \text{ is SR} \\ p(i, j, v) + f_P(v, B_i) + \alpha\tau_i(i, j, v) & (i, j, v) \text{ is RT} \\ p(i, j, v) + f_P(A_i, v) + \alpha\tau_i(i, j, v) & (i, j, v) \text{ is TR} \\ f_P(A_i, B_i) + \alpha\tau_i(i, j, v) & (i, j, v) \text{ is ST} \end{cases}$$

Each user's utility $\pi_i(i, j, v)$ is defined as the cost saving relative to traveling alone. A match $(i, j, v)$ is cost-feasible if

$$\begin{cases} \pi_i(i, j, v) = \theta_i^0 - \theta_i(i, j, v) \geqslant 0 \\ \pi_j(i, j, v) = \theta_j^0 - \theta_j(i, j, v) \geqslant 0 \end{cases} \tag{1}$$

All feasible matches are identified by enumerating candidate combinations and verifying their time and cost feasibility, with $M = M^{SR} \cup M^{RT} \cup M^{TR} \cup M^{ST}$ denoting the set of all feasible matches. For each user, define $M_i := \{(i, j, v) \in M : j \in D, v \in V\}$ and $M_j := \{(i, j, v) \in M : i \in R, v \in V\}$ as the sets of feasible matches involving rider $i$ and driver $j$, respectively.

## 2 Stable Matching Mechanism

MaaS is a user-centric system; here, users are autonomous and self-interested, aiming to maximize their own cost savings (Eq. (1)). Thus, the platform must consider users' incentives when determining a matching. Let $\mu$ denote a matching that maps $R \times D \times V$ onto itself, with selected matches denoted by $M_\mu$. For each match $(i, j, v) \in M$ selected by $\mu$, we write $\mu(i) = (j, v)$ for each $i \in R$ and $\mu(j) = (i, v)$ for each $j \in D$. If users remain unmatched, they would travel alone and receive zero cost savings.

### 2.1 Concept of stable matching

Following Wang et al.[7], we assume that user preferences depend solely on their cost savings. In practice, factors such as comfort and transfer convenience also matter. As an extension, these factors can be modeled as penalty terms and combined with cost savings to jointly determine preferences.

**Definition 1** User $i$ weakly prefers $(j, v)$ to $(j', v')$ in a matching, denoted as $(j, v) \succcurlyeq_i (j', v')$, if and only if (iff) $\pi_i(i, j, v) \geqslant \pi_i(i, j', v')$ in case $i \in R$ or $\pi_i(j, i, v) \geqslant \pi_i(j', i, v')$ in case $i \in D$.

Each user $i$ has a weak preference relation $\succcurlyeq_i$, possibly with ties. Users are assumed to be indifferent between matches yielding the same savings.

**Definition 2** A matching $\mu$ is stable iff there exists no match $(i, j, v) \in M \backslash M_\mu$ such that both $(j, v) \succ_i \mu(i)$ and $(i, v) \succ_j \mu(j)$. Such a match is referred to as a blocking match for $\mu$, where both users strictly prefer deviating from their current assignments.

In other words, a blocking match $(i, j, v)$ satisfies

$$\begin{cases} \pi_i(i, j, v) > \pi_i(i, \mu(i)) \\ \pi_j(i, j, v) > \pi_j(j, \mu(j)) \end{cases} \tag{2}$$

### 2.2 Stable matching model

The platform aims to compute a matching $\mu$ that is both feasible and stable. This can be formulated as an integer programming model. Let $x_{ijv} = 1$ if match $(i, j, v) \in M$ is selected in $\mu$, and 0 otherwise. The platform seeks to maximize the total revenue from MaaS operations.

$$(\text{MP}) \quad \Pi = \max \sum_{(i,j,v) \in M} \lambda p(i, j, v) x_{ijv}$$

$$\text{s. t.} \quad \sum_{(i,j,v) \in M_j} x_{ijv} \leqslant 1 \quad \forall j \in D \tag{3}$$

$$\sum_{(i,j,v) \in M_i} x_{ijv} \leqslant 1 \quad \forall i \in R \tag{4}$$

$$\sum_{\substack{(i',v') \succcurlyeq_j (i,v) \\ (i',j,v') \in M_j}} x_{i',j,v'} + \sum_{\substack{(j',v') \succcurlyeq_i (j,v) \\ (i,j',v') \in M_i}} x_{i,j',v'} + x_{ijv} \geqslant 1 \quad \forall (i, j, v) \in M \backslash M^{ST} \tag{5}$$

$$\sum_{\substack{(j',v') \succcurlyeq_i (0,d_i) \\ (i,j',v') \in M_i}} x_{i,j',v'} + x_{i,0,d_i} \geqslant 1 \quad \forall (i, 0, d_i) \in M^{ST} \tag{6}$$

$$x_{ijv} \in \{0, 1\} \quad \forall (i, j, v) \in M \tag{7}$$

Constraints (3) and (4) ensure that each user is assigned to at most one match. Constraint (5) requires that either driver $j$ or rider $i$ is matched to a more preferred option or that the match is assigned. Constraint (6) depends solely on rider preferences, as the transit system is not self-interested.

Stable matching may not be unique. When multiple stable matchings exist, the one that maximizes revenue is selected; alternative objectives (e. g., social welfare) can be used without affecting feasibility or stability. The final matching $\mu$ is induced by $\boldsymbol{x}$, with $M_\mu = \{(i, j, v) \in M : x_{ijv} = 1\}$.

## 3 Constraint-Generation Algorithm

As the number of users increases and the transit network size grows, the number of matches and the corresponding stability constraints grow rapidly. Stability constraints also become complex and require users' preference lists. Thus, it is time-consuming to directly solve MP by enumerating all constraints in practice. To address this, we develop an iterated stability searching (ISS) algorithm based on constraint generation. It iteratively solves a restricted MP (RMP) using a subset of stability constraints and then searches for blocking matches through a subproblem (SP). Any violated constraints are added incrementally until no further blocking matches can be found.

Let $\Theta \subset M \backslash M^{\mathrm{ST}}$ denote a subset of feasible no-ST matches. ISS initializes with $\Theta = \varnothing$ and iterates until no blocking match is found: (1) solve the RMP with set $\Theta$ of stability constraints; (2) update the temporary matching decision $\boldsymbol{x}$; (3) solve the constraint-generation SP to search for blocking matches; (4) add the blocking matches, if any, into set $\Theta$. Upon termination, the stable matching $\mu$ is induced by $\boldsymbol{x}$.

### 3.1 Restricted master problem

RMP includes stability constraint (6) for all ST matches while enforcing them for only a subset of non-ST matches in $\Theta$, making it easier to solve than the MP.

$$(\mathrm{RMP}) \quad \max \sum_{(i,j,v) \in M} \lambda p(i,j,v) x_{ijv}$$

$$\text{s. t.} \quad \sum_{(i,j,v) \in M_j} x_{ijv} \leqslant 1 \qquad \forall j \in D$$

$$\sum_{(i,j,v) \in M_i} x_{ijv} \leqslant 1 \qquad \forall i \in R$$

$$\sum_{\substack{(j',v') \geqslant_i (0,d_i) \\ (i,j',v') \in M_i}} x_{i,j',v'} + x_{i,0,d_i} \geqslant 1 \qquad \forall (i,0,d_i) \in M^{\mathrm{ST}}$$

$$\sum_{\substack{(i',v') \geqslant_j (i,v) \\ (i',j,v') \in M_j}} x_{i',j,v'} + \sum_{\substack{(j',v') \geqslant_i (j,v) \\ (i,j',v') \in M_i}} x_{i,j',v'} + x_{ijv} \geqslant 1$$

$$\forall (i,j,v) \in \Theta \backslash M^{\mathrm{ST}}$$

$$x_{ijv} \in \{0,1\} \qquad \forall (i,j,v) \in M$$

By solving RMP, we obtain a temporary matching $\mu$ that satisfies a subset $\Theta$ of stability constraints. Let $\Omega(\boldsymbol{x}) \coloneqq \{(i,j,v) \in M \backslash M^{\mathrm{ST}} : x_{ijv} = 0\}$ denote the set of unused non-ST matches under $\mu$. Blocking matches, if any, are searched from $\Omega(\boldsymbol{x})$ and must not belong to $\Theta$. Notably, RMP only requires preference lists for matches in $\Theta$, thereby significantly reducing computational complexity, and previously generated preference lists are retained across iterations.

### 3.2 Constraint-generation subproblem

To identify blocking matches, we formulate an SP based on stability condition (2). Let $\delta(i,j,v) = \pi_i(i,j,v) - \pi_i(i,\mu(i)) + \pi_j(i,j,v) - \pi_j(j,\mu(j))$ denote the value of the aggregated deviation. Define $z_{ijv} = 1$ if $(i,j,v)$ is selected as a blocking match, and 0 otherwise. The SP maximizes the total aggregated deviation values of the selected matches.

$$(\mathrm{SP}) \quad \max \sum_{(i,j,v) \in \Omega(\boldsymbol{x}) \backslash \Theta} \delta(i,j,v) z_{ijv}$$

$$\text{s. t.} \quad \sum_{(i,j,v) \in M_j \cap (\Omega(\boldsymbol{x}) \backslash \Theta)} z_{ijv} \leqslant 1 \qquad \forall j \in D \tag{8}$$

$$\sum_{(i,j,v) \in M_i \cap (\Omega(\boldsymbol{x}) \backslash \Theta)} z_{ijv} \leqslant 1 \qquad \forall i \in R \tag{9}$$

$$[\pi_i(i,j,v) - \pi_i(i,\mu(i))] z_{ijv} \geqslant 0 \qquad \forall (i,j,v) \in \Omega(\boldsymbol{x}) \backslash \Theta \tag{10}$$

$$[\pi_j(i,j,v) - \pi_j(j,\mu(j))] z_{ijv} \geqslant 0 \qquad \forall (i,j,v) \in \Omega(\boldsymbol{x}) \backslash \Theta \tag{11}$$

$$z_{ijv} \in \{0,1\} \qquad \forall (i,j,v) \in \Omega(\boldsymbol{x}) \backslash \Theta \tag{12}$$

Constraints (8) and (9) ensure that each user is involved in at most one blocking match, whereas constraints (10) and (11) enforce blocking conditions. If the optimal value of SP is positive, each $(i,j,v)$ with $z_{ijv} = 1$ is identified as a blocking match and added to $\Theta$. In the next iteration, RMP includes the corresponding stability constraints for these newly added matches. Otherwise, ISS stops.

SP does not require user preference lists and is easier to solve. Because many stability constraints in MP are redundant, adding too many blocking matches at once increases the computational burden of RMP. Therefore, constraints (8) and (9) are necessary to limit them.

## 4 Computational Experiments

### 4.1 Chengdu case

Our experiments used real ride-request data from Didi Chuxing[19] in Chengdu, China. The dataset recorded pickup/drop-off times and locations in November 2016. Here, 7 500 morning-commute orders were sampled to generate instances. The subway network included eight lines and 284 stations. Travel distances, times, and fares for taxi and subway trips were obtained through the AMap API[20]. The walking distance limit $\bar{W}$ was 0.8 km. Ridesharing fares followed the pricing scheme of Dida Chuxing[18].

Orders were randomly selected and assigned as drivers or riders. For order $i$, the earliest departure $T_i^-$ was set $\varepsilon_{\mathrm{p}}$ (unit: min) before pickup time; the maximum duration $T_i^{\max}$ was $(1+\varepsilon_{\mathrm{t}}^{\mathrm{r}})$ or $(1+\varepsilon_{\mathrm{t}}^{\mathrm{d}})$) times the travel-alone time; and the latest arrival $T_i^+ = T_i^- + T_i^{\max} + \varepsilon_{\mathrm{p}}$. Here,

$\varepsilon_p$ and $(\varepsilon_t^d, \varepsilon_t^r)$ captured the departure time and duration flexibility, respectively. We considered five factors when generating an instance: (1) $|R|/|D| = 1$ or 2; (2) $\varepsilon_p = 20$ or 10 min; (3) $(\varepsilon_t^d, \varepsilon_t^r) = (40\%, 20\%)$ or $(60\%, 30\%)$; (4) $\alpha = 15$ or 30 yuan/h; (5) commission fee scheme, 10% flat rate or the distance-based scheme in Dida Chuxing[18]. The first level of each factor served as the baseline.

All algorithms were implemented in Python 3.11 and solved using Gurobi v12.0.0. Experiments were run on a desktop with an Intel® Core (TM) i9-9900 CPU (3.10 GHz) and 64 GB of RAM.

### 4.2 Computational efficiency analysis

We evaluated computational efficiency by comparing ISS with two enumeration-based algorithms. E_all solved MP by enumerating stability constraints (5) and (6), whereas E_iter followed the ISS framework but added all blocking matches each iteration. We generated 10 small-scale instances with $|U| \in [1\,000, 6\,000]$ and tested all three algorithms (Table 1). For large-scale instances with $|U| \in [6\,000, 7\,500]$, only ISS was tested because of scalability limits (Table 2).

**Table 1** Algorithm efficiency performance on small-scale instances

| Instance | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Basic data | $\|U\|$ | 1 000 | 1 500 | 2 000 | 2 500 | 3 500 | 4 000 | 4 500 | 5 000 | 5 500 | 6 000 |
| | $\|M\|/10^4$ | 2.44 | 4.43 | 8.37 | 14.17 | 26.51 | 36.21 | 45.28 | 55.49 | 63.36 | 78.53 |
| | $T_f$/min | 0.05 | 0.09 | 0.17 | 0.28 | 0.55 | 0.72 | 0.92 | 1.16 | 1.81 | 1.88 |
| ISS | $\Pi/10^3$ yuan | 0.23 | 0.33 | 0.40 | 0.53 | 0.72 | 0.87 | 0.97 | 1.08 | 1.16 | 1.31 |
| | $N_{iter}$ | 10 | 19 | 14 | 19 | 26 | 21 | 26 | 29 | 25 | 22 |
| | $T_{alg}$/min | 0.05 | 0.21 | 0.31 | 1.16 | 5.15 | 3.46 | 11.18 | 14.2 | 18.3 | 10.71 |
| | $N_{con}/10^4$ | 0.03 | 0.07 | 0.09 | 0.15 | 0.24 | 0.28 | 0.36 | 0.40 | 0.44 | 0.48 |
| E_iter | $\Pi/10^3$ yuan | 0.23 | 0.33 | 0.40 | 0.53 | 0.72 | 0.87 | 0.97 | 1.08 | 1.16 | 1.31 |
| | $N_{iter}$ | 10 | 17 | 8 | 15 | 24 | 15 | 18 | 18 | 20 | 27 |
| | $T_{alg}$/min | 0.16 | 1.28 | 2.52 | 9.94 | 106.59 | 118.51 | 350.85 | 638.03 | 982.59 | 1894.91 |
| | $N_{con}/10^4$ | 0.23 | 0.64 | 1.35 | 1.86 | 3.76 | 5.26 | 6.69 | 8.49 | 9.13 | 12.2 |
| E_all | $\Pi/10^3$ yuan | 0.23 | 0.33 | 0.40 | 0.53 | 0.72 | 0.87 | 0.97 | 1.08 | 1.16 | 1.31 |
| | $T_{alg}$/min | 0.45 | 0.94 | 4.3 | 8.08 | 36.66 | 55.51 | 87.54 | 154.85 | 120.8 | 262.46 |
| | $N_{con}/10^4$ | 2.41 | 4.40 | 8.32 | 14.11 | 26.42 | 36.11 | 45.17 | 55.38 | 63.22 | 78.38 |

**Table 2** Algorithm efficiency performance on large-scale instances

| Instance | | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Basic data | $\|U\|$ | 6 302 | 6 719 | 6 742 | 6 769 | 6 146 | 6 902 | 6 919 | 6 686 | 7 079 | 6 191 |
| | $\|M\|/10^4$ | 88.13 | 97.68 | 100.72 | 102.40 | 82.52 | 103.31 | 104.27 | 97.61 | 109.58 | 80.90 |
| | $T_f$/min | 2.28 | 2.47 | 2.51 | 2.61 | 2.13 | 2.69 | 2.67 | 2.49 | 2.84 | 2.14 |
| ISS | $\Pi/10^3$ yuan | 3.45 | 3.67 | 3.65 | 3.67 | 3.30 | 3.73 | 3.81 | 3.63 | 3.87 | 3.31 |
| | $N_{iter}$ | 21 | 33 | 24 | 23 | 26 | 25 | 25 | 22 | 26 | 24 |
| | $T_{alg}$/min | 14.49 | 31.5 | 15.15 | 25.58 | 15.84 | 27.67 | 22.84 | 17.2 | 32.96 | 18.67 |
| | $N_{con}/10^4$ | 0.54 | 0.60 | 0.59 | 0.60 | 0.53 | 0.60 | 0.62 | 0.58 | 0.64 | 0.51 |

Feasible matches were generated quickly. All algorithms yielded the same objective value ($\Pi$), confirming that ISS preserves optimality. ISS showed a short computation time ($T_{alg}$) and required a few iterations ($N_{iter}$). It is generally six to ten times faster than E_all and significantly outperforms E_iter. When $|U| > 2\,500$, E_iter became much slower than E_all, highlighting the advantage of the SP formulation. On very large instances, ISS still exceeded 30 min as the RMP size grew. ISS also generated far fewer stability constraints ($N_{con}$), about 1% of those in E_all.

### 4.3 Stability analysis

We considered five factors at two levels (32 settings). For each setting, we generated five random instances with $|U| \in [2\,000, 4\,000]$ (160 total) and compared stable matching $\mu^s$ with centralized matching $\mu^c$, where $\mu^c$ was obtained by solving the MP without stability constraints (5) and (6).

Under matching $\mu^k$ with $k \in \{c, s\}$, we assessed the following six metrics: (1) $\phi_{XT}^k$ ($\phi_{SR}^k$), share of RT/TR (SR) among the selected matches; (2) $\rho_r^k$ ($\rho_d^k$), fraction of matched riders (drivers) among all riders (drivers); (3) $\zeta_r^k$ ($\zeta_d^k$), average relative travel time increase versus traveling alone for riders (drivers) assigned to non-ST matches; (4) $\chi_r^k$ ($\chi_d^k$), average relative cost savings versus traveling alone; (5) $\Pi_{loss} = 100(\Pi^c - \Pi^s)/\Pi^c$, where $\Pi^c$ and $\Pi^s$ denote platform revenues under $\mu^c$ and $\mu^s$, respectively; (6) $b_p$, share of selected matches containing at least one user in a blocking match.

Fig. 2 reports user-level outcomes. In Fig. 2(a), compared with $\mu^c$, $\mu^s$ selected more RT/TR and slightly

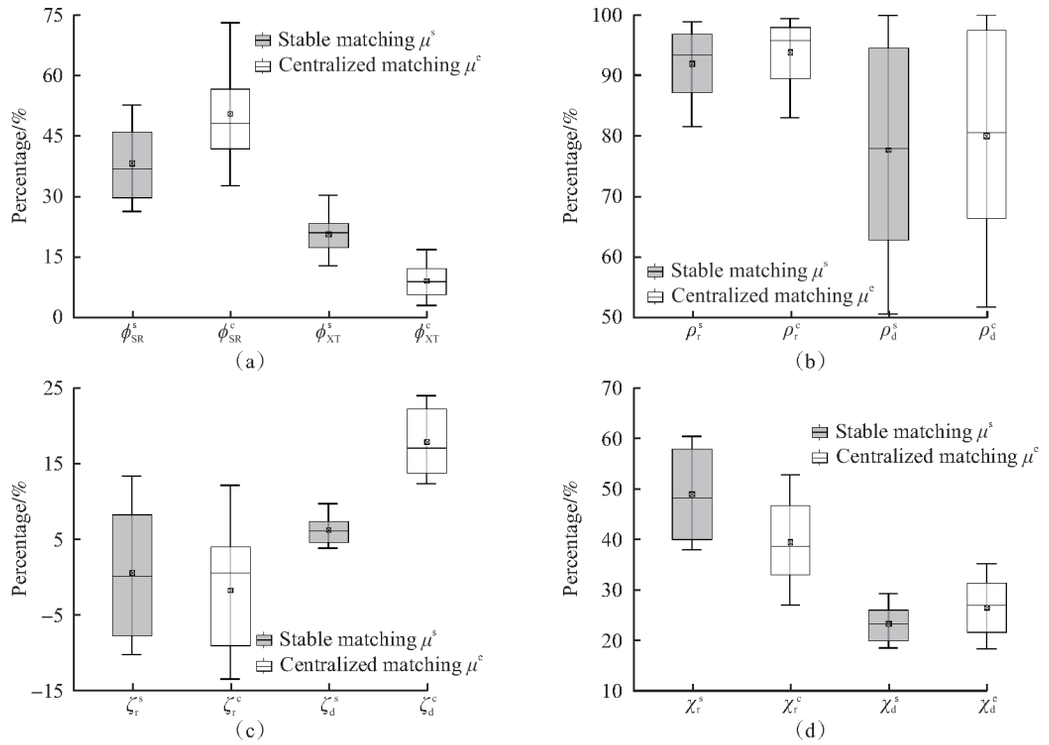


**Fig. 2** Comparison of user benefits. (a) Match types; (b) Matching rates; (c) Travel detour; (d) Cost savings

fewer SR matches because SR matches yielded higher revenue and were favored by $\mu^c$. In Fig. 2(b), the matching rates ($\rho_d$, $\rho_r$) were similar, with $\mu^s$ being slightly lower on average. $\rho_r$ typically exceeded 85% because of transit use, whereas $\rho_d$ usually ranged from 60% to 90%, depending on the rider-driver ratio. Fig. 2(c) shows that $\mu^s$ reduced driver detours but increased rider detours because the greater use of RT/TR lengthened riders' trips while drivers' routes changed little. In Fig. 2(d), riders obtained higher cost savings under $\mu^s$ because of lower payments but drivers obtained less, consistent with shorter shared distances.

Fig. 3 compares revenue and stability. In Fig. 3(a), the mean revenue loss ($\Pi_{loss}$) is 31.43%, indicating substantial revenue loss under $\mu^s$ due to the greater use of RT/TR matches, consistent with Fig. 2(a). In Fig. 3(b), the blocking match ratio ($b_p$) often exceeds 35% and can reach up to 75%. This indicates that $\mu^c$ is highly unstable and is likely unacceptable to most users. To balance revenue and stability, the platform may consider a relaxed notion of $\varepsilon$-stability and compute nearly stable matchings.

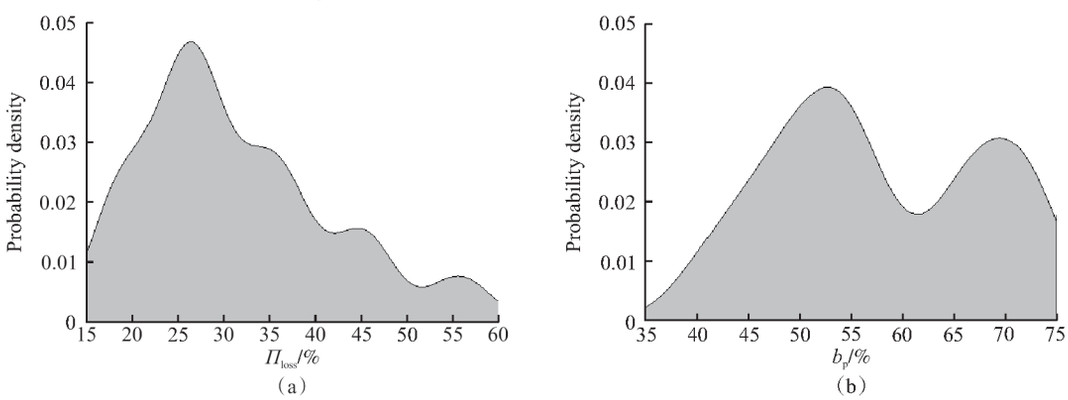


**Fig. 3** Revenue and stability comparison. (a) Revenue loss; (b) Blocking ratio

## 5 Conclusions

(1) This study examined the operation of a MaaS platform integrating ridesharing and public transit. The integrated matching problem was formulated as a two-sided one-to-one stable matching model with numerous stability constraints. To solve it efficiently, an iterated constraint-generation algorithm with a tractable SP was proposed without requiring explicit preference lists.

(2) Extensive experiments demonstrated the efficiency of our algorithm without compromising solution quality. Compared with centralized matching, stable matching selected more RT/TR matches, leading to longer detours and higher cost savings for riders; meanwhile, the oppo-

site held for drivers. Despite some revenue loss, stable matching increased transit usage and was more acceptable to users, which was vital for system sustainability.

(3) The analysis assumed an exogenous ridesharing payment rule. A key direction for future research is to design incentives that align stable matchings with system-optimal outcomes or improve revenue under stability, encouraging users to accept socially optimal matches.

## References


[1] JITTRAPIROM P, CAIATI V, FENERI A M, et al. Mobility as a service: A critical review of definitions, assessments of schemes, and key challenges [J]. Urban Planning, 2017, 2(2): 13-25.

[2] HEIKKILÄ S. Mobility as a service: A proposal for action for the public administration, case Helsinki [D]. Helsinki, Finland: Aalto University, 2014.

[3] WRIGHT S, NELSON J D, COTTRILL C D. MaaS for the suburban market: Incorporating carpooling in the mix [J]. Transportation Research Part A: Policy and Practice, 2020, 131: 206-218.

[4] WANG Q C, JANNICKE H B, SEBASTIAAN M. The complexity of stakeholder influence on MaaS: A study on multi-stakeholder perspectives in Shenzhen self-driving mini-bus case [J]. Research in Transportation Economics, 2022, 94: 101070.

[5] SHEN J X, LI Q N, YU M, et al. Group effect analysis of factors influencing subway passenger satisfaction from a travel chain perspective [J]. Journal of Southeast University (Natural Science Edition), 2025, 55(4): 1165-1172. (in Chinese)

[6] LU R Y, GUO X C, LI J C, et al. Tourist travel behavior in rural areas considering bus route preferences [J]. Journal of Southeast University (English Edition), 2023, 39(1).

[7] WANG X, AGATZ N, ERERA A. Stable matching for dynamic ride-sharing systems [J]. Transportation Science, 2017, 52(4): 850-867.

[8] SONG X Q, ZHANG W J, LEI L, et al. Dynamic resource allocation algorithm for multi-objective joint optimization in Internet of vehicle [J]. 2025, 55(1): 266-274. (in Chinese)

[9] UTRIAINEN R, PÖLLÄNEN M. Review on mobility as a service in scientific publications [J]. Research in Transportation Business & Management, 2018, 27: 15-23.

[10] LYONS G, HAMMOND P, MACKAY K. The importance of user perspective in the evolution of MaaS [J]. Transportation Research Part A: Policy and Practice, 2019, 121: 22-36.

[11] STIGLIC M, AGATZ N, SAVELSBERGH M, et al. Enhancing urban mobility: Integrating ride-sharing and public transit [J]. Computers & Operations Research, 2018, 90: 12-21.

[12] KUMAR P, KHANI A. An algorithm for integrating peer-to-peer ridesharing and schedule-based transit system for first mile/last mile access [J]. Transportation Research Part C: Emerging Technologies, 2021, 122: 102891.

[13] GALE D, SHAPLEY L S. College admissions and the stability of marriage [J]. The American Mathematical Monthly, 1962, 69(1): 9-15.

[14] ROTH A E, SOTOMAYOR M. Chapter 16 two-sided matching [M]. Amsterdam, the Netherlands: Elsevier, 1992: 485-541.

[15] VANDE VATE J H. Linear programming brings marital bliss [J]. Operations Research Letters, 1989, 8(3): 147-153.

[16] ROTHBLUM U G. Characterization of stable matchings as extreme points of a polytope [J]. Mathematical Programming, 1992, 54(1): 57-67.

[17] FENG S Y, DUAN P B, KE J T, et al. Coordinating ride-sourcing and public transport services with a reinforcement learning approach [J]. Transportation Research Part C: Emerging Technologies, 2022, 138: 103611.

[18] Dida Chuxing. The convention rules of Dida Chuxing [EB/OL]. (2024-12-24) [2025-05-24]. http://www.didachuxing.com/homepage/carpool/convention. (in Chinese)

[19] Didi Chuxing. KDD Cup 2020 dataset [EB/OL]. (2020-06-11) [2025-05-24]. https://outreach.didichuxing.com/app-vue/KDD_CUP_2020 id=1004. (in Chinese)

[20] AMap. Route planning API document [EB/OL]. (2025-02-14) [2025-05-24]. https://lbs.amap.com/api/webservice/guide/api/direction. (in Chinese)


# 出行即服务平台的多模式交通整合稳定匹配机制

吴雅婷, 赖明辉

(东南大学经济管理学院,南京 211189)

**摘要:** 在拼车与公共交通整合的出行即服务平台中,乘客-司机对可结合不同换乘站点形成多个潜在匹配方案,且用户为自利异质偏好主体。在生成所有的可行整合匹配之后,以平台收益最大化为目标,构建了双边一对一稳定匹配模型,其中每个可行匹配对应一个嵌套偏好信息的稳定性约束。为高效求解该模型,提出了一种迭代约束生成算法。该算法重复求解一个受限主问题来获得暂时解和子问题,从而识别违反的稳定性约束。实验结果表明:所提算法可显著提高计算效率;与阻塞率高达75%的集中式匹配相比,稳定匹配虽使平台收入平均降低31.43%,但能显著提高公共交通使用率与用户接受度,且乘客绕行距离更长,费用节省更明显,司机则相反。

**关键词:** 出行即服务;稳定匹配;约束生成;拼车;公共交通

**中图分类号:** C934;U121;U15